HITEC ROBOTICS

ROBONOVA-I



Bedienungsanleitung

Sicherheitshinweise

HITEC/MULTIPLEX – ROBONOVA-I –Baukästen/RTW unterliegen während der Produktion einer ständigen Materialkontrolle. Wir hoffen, dass Sie mit dem Inhalt zufrieden sind. Wir bitten Sie, den Inhalt vor Verwendung zu prüfen. Sie bekommen als unsere Kunden automatisch den Email Helpdesk Service und es ist der schnellste Weg mit uns Kontakt aufzunehmen. Bei Fragen schauen Sie zuerst im Handbuch nach und im Internetbereich Support finden Sie diverse Antworten. Wenn Sie keine Lösung finden, kontaktieren Sie uns.

Wir arbeiten ständig an der technischen Weiterentwicklung unserer Modelle. Änderungen des Packungsinhalts in Form, Maß, Technik, Material und Ausstattung behalten wir uns jederzeit und ohne Ankündigung vor. Bitte haben Sie Verständnis dafür, dass aus Angaben und Abbildungen dieser Anleitung keine Ansprüche abgeleitet werden können.

Diese Bedienungsanleitung ist Bestandteil des Produktes. Sie beinhaltet wichtige Informationen und Sicherheitshinweise. Sie ist deshalb jederzeit griffbereit aufzubewahren und beim Verkauf des Produktes an Dritte weiterzugeben. ! Sicherheitshinweise beachten!

Anleitung sorgfältig lesen!
Gerät nicht in Betrieb nehmen, bevor Sie diese Bedienungsanleitung und die folgenden (bzw. in der Anleitung enthaltenen oder separat beiliegenden) Sicherheitshinweise sorgfältig gelesen haben.
Nehmen Sie unter keinen Umständen technische Veränderungen an dem Gerät vor. Verwenden Sie ausschließlich Original-Zubehör- und Ersatzteile Gerät bzw. Modell nicht in Betrieb nehmen, wenn etwas nicht in Ordnung scheint. Zuerst Fehler suchen und beheben.

- Das Gerät ist für Kinder unter 14 Jahren nicht geeignet.

- Durch die scharfen Kanten der Aluminiumverbindungsteile können bei unsachgemäßer Handhabung Verletzungen auftreten.

- Bei Betrieb des Roboters können durch unvorhergesehene Bewegungsabläufe Finger eingeklemmt werden. Daher sollte der Roboter während des Betriebes nicht in die Hand genommen werden. Es besteht Quetschgefahr an Servos und Metallteilen. Zum Testlauf unbedingt Halterung verwenden

- Bei Wartungs- und Servicearbeiten die Stromversorgung (Akku) abtrennen.

- Bei Montage und Service ist große Sorgfalt erforderlich. Die Servos dürfen in keinem Betriebsfalle gegen mechanische Anschläge laufen können. Bei der Montage ist darauf zu achten, dass die Servokabel sorgfältig verlegt werden und keinesfalls von der Mechanik eingeklemmt werden können. Bei Beschädigung des Kabels besteht Kurzschluss- und eventuell sogar Brandgefahr.

- Aus Sicherheitsgründen sollte der Roboter nur auf ebenen Oberflächen betrieben werden.

ESD-Hinweise für elektronische Baugruppen:
Die Baugruppen Ihres Gerätes sind mit elektrostatisch empfindlichen Bauteilen bestückt. Diese können durch Ladungsausgleich (Potentialausgleich durch elektrostatische Entladung) beim Berühren der Baugruppe zerstört oder in der Lebensdauer beeinflusst werden.

Beachten Sie unbedingt folgende Schutzmaßnahmen für elektrostatisch gefährdete Baugruppen:

- Stellen Sie vor dem Einsetzen bzw. Ausbau solcher Baugruppen in das Gerät einen elektrischen Potentialausgleich zwischen sich und ihrer Umgebung her (z.B. Heizkörper anfassen). Öffnen Sie ggf. das Gerät und fassen es großflächig an, um den Potentialausgleich zum Grundgerät zu schaffen.
- Nehmen Sie Baugruppen erst nach dem Potentialausgleich aus dem leitfähigen ESD-Schutzbeutel heraus. Vermeiden Sie die direkte Berührung von elektronischen Bauteilen oder Lötpunkten. Fassen Sie die Baugruppe nur am Rand der Platine.

Gewährleistung
Für unsere Produkte leisten wir entsprechend den derzeit geltenden gesetzlichen Bestimmungen Gewähr. Wenden Sie sich mit Gewährleistungsfällen an den Fachhändler, bei dem Sie das Gerät erworben haben. Von der Gewährleistung ausgeschlossen sind Fehlfunktionen, die verursacht wurden durch:

- unsachgemäßen Betrieb.
- durch falsche, nicht oder verspätet, oder nicht von einer autorisierten Stelle durchgeführte Wartung.
- falsche Anschlüsse.
- Verwendung von nicht originalem HITEC/MULTIPLEX Zubehör,
- Veränderungen/Reparaturen, die nicht von MULTIPLEX oder einer MUTIPLEX- Service-Stelle ausgeführt wurden.
- versehentliche oder absichtliche Beschädigungen.
- Defekte, die sich aus der normalen Abnutzung ergeben.

Betrieb außerhalb der technischen Spezifikationen, oder im Zusammenhang mit Geräten anderer Hersteller

Haftung / Schadenersatz
Sowohl die Einhaltung der Hinweise aus Montage- und Bedienungsanleitung, als auch die Bedingungen und Methoden bei Installation, Betrieb, Verwendung und Wartung des Gerätes können von der Firma MULTIPLEX Modellsport GmbH & Co. KG nicht überwacht werden. Daher übernimmt die Firma MULTIPLEX Modellsport GmbH & Co. KG keinerlei Haftung für Verluste, Schäden oder Kosten, die sich aus fehlerhafter Verwendung und Betrieb ergeben oder in irgendeiner Weise damit zusammenhängen. Soweit gesetzlich zulässig, ist die Verpflichtung der Firma MULTIPLEX Modellsport GmbH & Co. KG zur Leistung von Schadenersatz, gleich aus welchem Rechtsgrund, begrenzt auf den Rechnungswert der an dem schadenstiftenden Ereignis unmittelbar beteiligten Warenmenge der Firma MULTIPLEX Modellsport GmbH & Co. KG. Dies gilt nicht, soweit die Firma MULTIPLEX Modellsport GmbH & Co. KG nach zwingenden gesetzlichen Vorschriften wegen Vorsatzes oder grober Fahrlässigkeit unbeschränkt haftet.

CE-Konformitätserklärung
Die Firma MULTIPLEX Modellsport GmbH & Co. KG erklärt für **ROBONOVA-I** die Übereinstimmung mit folgenden harmonisierten Richtlinien der EU:
Schutzanforderungen in Bezug auf die elektromagnetische Verträglichkeit
Protection requirements concerning electromagnetic Compatibility

EN55014-1
EN55014-2

Sicherheitshinweise zum Thema Akku:

- Akkus müssen außerhalb der Reichweite von Kindern aufbewahrt werden.
- Vor jedem Gebrauch den einwandfreien Zustand des Akkus prüfen. Beschädigte oder defekte Akkus nicht mehr verwenden.
- Akkus nicht erhitzen, verbrennen, öffnen, kurzschließen, mit überhöhten Strömen laden oder entladen, überladen oder tiefentladen, verpolt laden.
- Akkus während des Ladevorgangs außerhalb des Gerätes auf eine hitzebeständige, nicht brennbare und nicht leitende Unterlage legen und nicht unbeaufsichtigt lassen.
- Keine Veränderungen an Akku-Packs vornehmen. Niemals direkt an den Zellen löten oder schweißen.
- Bei falscher Behandlung besteht Entzündungs-, Explosions-, Verätzungs-, und Verbrennungsgefahr.
  Geeignete Löschmittel: Wasser, CO_2, Sand
- Auslaufendes Elektrolyt ist ätzend!
  Nicht mit Haut oder Augen in Berührung bringen.
  Im Notfall sofort mit reichlich Wasser ausspülen und einen Arzt aufsuchen.
- Verbrauchte Akkus ordnungsgemäß nach den jeweiligen gesetzlichen Bestimmungen entsorgen, keinesfalls in den normalen Hausmüll geben.

ROBONOVA-I Montageanleitung

1 Allgemeine Hinweise

1.1 Stückliste

130000	130001	130002	130003	130004
U - Blech mit Gewinde HR1B-0001	U - Blech ohne Gewinde HR1B-0002	I - Blech HR1B-0003	H - Blech HR1B-0004	Körper Rückseite HR1B-0005
6 Stück	4Stück	8 Stück	2 Stück	1 Stück

130005	130006	130007	130008	130009
Körper Vorderteil HR1B-0006	Körper Oberteil HR1B-0007	Körper Unterteil HR1B-0008	Schulter Innenblech HR1B-0009	Schulter Außenblech HR1B-00010
1 Stück	1 Stück	1 Stück	2 Stück	2 Stück

130010	130011	131000	131001	131002
Hand Verbinder HR1B-00011	Fuß HR1B-00012	Abdeckung vorne HR1C-0001	Abdeckung hinten HR1C-0002	Visier HR1C-0003
2 Stück	2 Stück	1 Stück	1 Stück	1 Stück

131003	131004	131005	131006	131008
Kopf Vorderteil HR1C-0004	Kopf Hinterteil HR1C-0005	Hand-Hälfte HR1C-0006	Hand-Hälfte HR1C-0007	Fußabdeckung, rechts HR1C-0008
1 Stück	1 Stück	2 Stück	2 Stück	1 Stück

131009	132001	132002	132003	132004
Fußabdeckung, linksHR1C-0009	Unterlegscheibe, 6x2,2x0,5mm, Ni	Schraube, M2 x 4mm, Ni	Schraube, M2,6 x 4mm, Ni	Schraube, M3 x 4mm
1 Stück	28 Stück	40 Stück	12 Stück	4 Stück

132005	132006	132007	132008	132009
Abstandsbolzen mit Gewinde M3	Rändelschraube M3	Schraube, selbstschneidend, 2 x 4mm, Ni	Schraube, selbstschneidend, 2 x 5mm, Ni	Schraube, selbstschneidend, 2 x 8mm, Ni
4 Stück	2 Stück	128 Stück	12 Stück	6 Stück

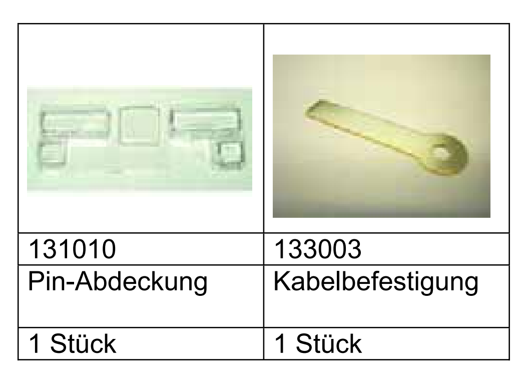
132013	133000	133001	133002	134000
Schraube, selbstschneidend, 2 x 26mm, sw	Kabelbinder	Kabelbefestigung	Akkukabel Schutz	Servo Nr.1 HSR-8498HB1R200
2 Stück	8 Stück	28 Stück	1 Stück	1 Stück

134001	134002	134003	134004	134005
Servo Nr.3 HSR-8498HB1L200	Servo Nr.2 HSR-8498HB1R300	Servo Nr.4 HSR-8498HB1L300	Servo Nr.5 HSR-8498HB2R200	Servo Nr.7 HSR-8498HB2L200
1 Stück	3 Stück	3 Stück	1 Stück	1 Stück

134006	134007	134008	134009	134010
Servo Nr.6 HSR- 8498HB2R400	Servo Nr.8 HSR- 8498HB2L400	Servo Nr.9 HSR- 8498HB3R200	Servo Nr.10 HSR- 8498HB3L200	Servo Nr.11 HSR- 8498HB2R300
1 Stück	1 Stück	1 Stück	1 Stück	1 Stück

134011	135004	136000	136001	136002
Servo Nr.12 HSR- 8498HB2L300	Servogegenlager 4,5mm	Controller MR-C3024	LED Platine	IR-Sensor
1 Stück	1 Stück	1 Stück	1 Stück	1 Stück

136003	136010	136020	136030	136040
Remocon IR-Fernbedienung	Akku 6V / 1000mAh	CD-ROM (ROBOBASIC V2. & ROBONOVA-I Anleitung	PC-Kabel seriell MR-C3024	Schnellladegerät
1 Stück	1 Stück	1 Stück	1 Stück	1 Stück



131010	133003
Pin-Abdeckung	Kabelbefestigung
1 Stück	1 Stück

1.2 Anmerkung zu den Servos

Spezifikationen:
Interface : HMI Protokoll, PWM
Betriebsspannung: 6.0V
Stellgeschwindigkeit: 0.20sec/60°bei 6.0V
Max. Drehmoment: 100Ncm bei 6.0V
Drehwinkel: Max 180°
Gewicht: 55g
Dimension: 40 x 20 x 47mm

Impulsspezifikationen:
Neutral : 1500 ms
0-180°: ±1100-1900 ms
Wiederholrate: 12-26ms

Die HSR-8498HB Servos sind auf den jeweiligen Einsatzort abgestimmt.
Sie unterscheiden sich in Gehäuse, Abtriebsrad, Kabellänge und Einbaurichtung
Um die Servos schnell zu identifizieren sind sie mit Nummern versehen.
Halten Sie sich bei der Montage genau an die Anweisungen, bzw. orientieren Sie sich an den Bildern. Somit stellen Sie sicher, dass das richtige Servo an der richtigen Stelle eingesetzt wird.
Sie müssen keine Anpassungen an der Kabellänge durchführen

Bei der Montage der Servos werden teilweise die Schrauben, die das Servo zusammenhalten entfernt, um die verschiedenen Bleche zu befestigen.
Achten sie unbedingt darauf, dass die unterschiedlich langen Schrauben an der richtigen Stelle wieder eingeschraubt werden, da sonst die Elektronik oder die Mechanik im Servo beschädigt werden kann.

Die folgenden Servos kommen bei ROBONOVA-I zum Einsatz:
Nr. 1, HSR-8498HB 1R 200 (1 Stück)
Nr. 2, HSR-8498HB 1R 300 (3 Stück)
Nr. 3, HSR-8498HB 1L 200 (1 Stück)
Nr. 4, HSR-8498HB 1L 300 (3 Stück)
Nr. 5, HSR-8498HB 2R 200 (1 Stück)
Nr. 6, HSR-8498HB 2R 400 (1 Stück)
Nr. 7, HSR-8498HB 2L 200 (1 Stück)
Nr. 8, HSR-8498HB 2L 400 (1 Stück)
Nr. 9, HSR-8498HB 3R 200 (1 Stück)
Nr. 10, HSR-8498HB 3L 200 (1 Stück)
Nr. 11, HSR-8498HB 2R 300 (1 Stück)
Nr. 12, HSR-8498HB 2L 300 (1 Stück)

1.3 Was wird für die Montage benötigt?

Sie benötigen einen Kreuzschlitzschraubendreher der Größen PH0 und PH1, möglichst magnetisch, eine spitze Pinzette, bzw. eine Spitzzange, sowie ein Gabel- oder Steckschlüssel SW5 (alternativ funktioniert auch eine Flachzange).
Achten Sie auf eine gute Werkzeugqualität um Schrauben und Bauteile nicht zu beschädigen.

1.4 Links oder rechts?

Wenn im Verlauf der folgenden Anleitung vom „rechten Arm“ oder vom „linken Fuß“ die Rede ist, dann ist dies immer aus der Sicht des Roboters zu sehen.

1.5 Tipps zum Akku

Damit Sie nach erfolgreicher Montage sofort mit der Inbetriebnahme weitermachen können, sollte als erstes der Akku geladen werden. Da eine direkte Verbindung von Akku und Ladegerät nicht vorgesehen ist, muss der Akku (136010) mit der Controller Leiterplatte MR-C3024 (136000) verbunden werden. Anschließend stecken sie das Schnellladegerät (136040) in die Ladebuchse auf den Controller Leiterplatte. Legen Sie den Akku und die Leiterplatte auf eine elektrisch nicht leitende Oberfläche und Stecken Sie das Ladegerät in die Steckdose.

2 Montage der Einzelteile

2.1 Aufbau der Beine

2.1.1 Linker und rechter Fuß:

Benötigte Bauteile:

- zwei Fußteile HR1B-00012 (130011)
- ein Servo Nr.6 (134007, linker Fuß)
- ein Servo Nr.8 (134006, rechter Fuß)

Montieren Sie die Servos so, wie auf den Bildern zu sehen ist (Das Servokabel auf der Oberseite, achten Sie auf die freien Löcher in den Fußteilen). Dazu müssen sie zuerst die Abtriebsräder sowie die Gegenlager von den Servos entfernen und die unteren Schrauben (2x schwarz, 2x silbern) aus den Servos herausdrehen.
Positionieren Sie nun die Servos im Fußteil und drehen Sie die Schrauben wieder vorsichtig hinein (Achten Sie auf die korrekte Position der kürzeren silbernen Schrauben und der längeren schwarzen Schrauben).
Anschließend montieren Sie auch wieder die Abtriebsräder sowie die Gegenlager. Die Abtriebsräder lassen sich wegen dem Positionierungsstift nur in einer Position auf den Vielzahn stecken.
Vor ihnen müssen nun zwei symmetrische Füße stehen.







2.1.2 *Verbindungen Sprunggelenk und Oberschenkel:*

Benötigte Bauteile:

- vier U-Bleche mit Gewinde HR1B-0001 (130000)
- vier U-Bleche ohne Gewinde HR1B-0002 (130001)
- 16 Schrauben M2 x 4mm (132002)

Achtung: die beiden U-Bleche sind sich sehr ähnlich. Neben dem Hauptunterschied, den Gewinden, ist bei dem U-Blech mit Gewinde HR1B-0001die seitlich hochgebogene Kante höher!
Verschrauben Sie jeweils ein Blech vom Typ HR1B-0001 mit einem Blech HR1B-0002, wie im Bild zu sehen

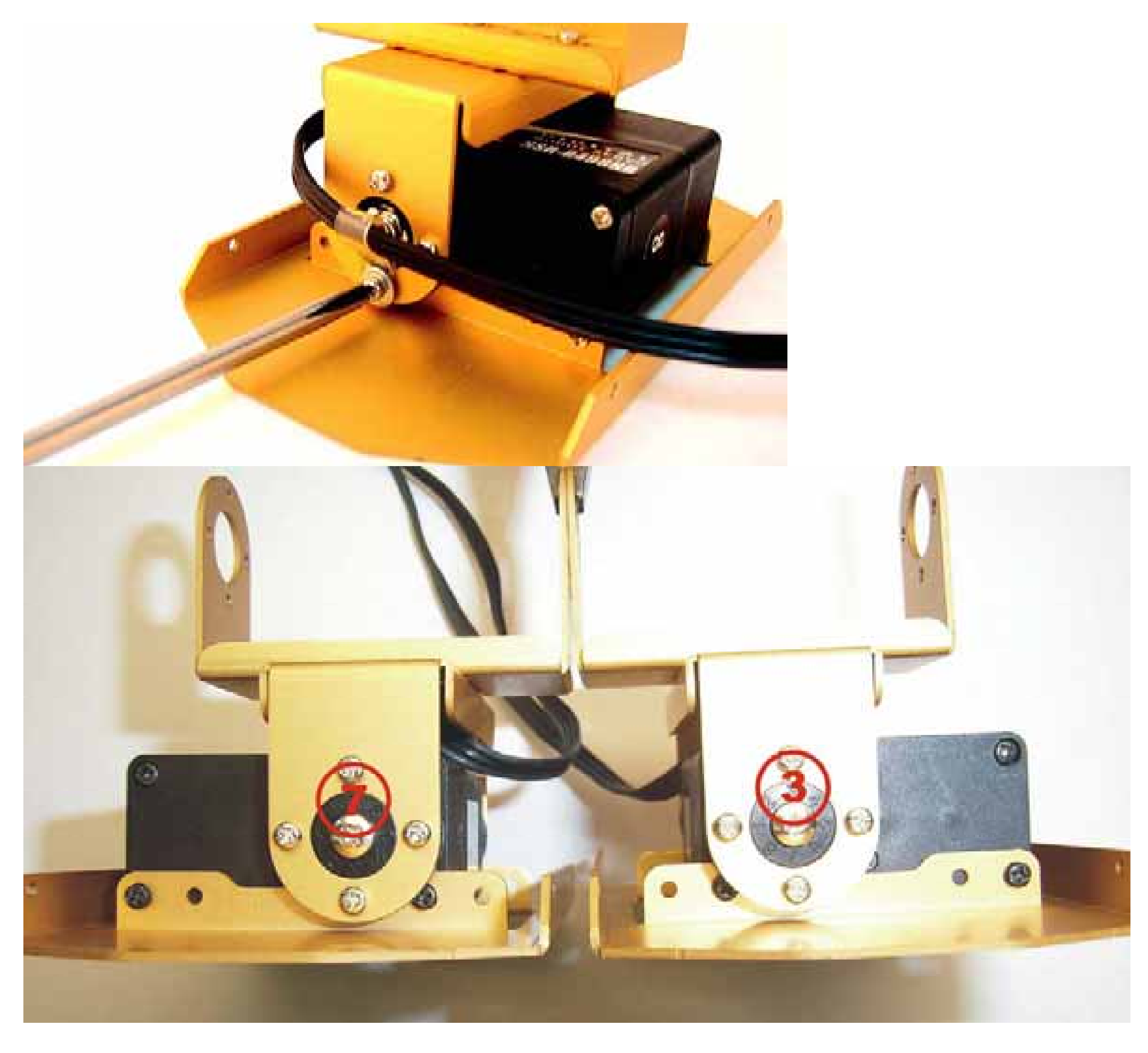


2.1.3 Sprunggelenk:

Benötigte Teile:

- 2 Fußbaugruppen aus *Schritt 2.1.1*
- 2 Sprunggelenksverbindungen aus *Schritt 2.1.2*
- 16 Schrauben 2 x 4mm (132007)
- 2 Kabelbefestigungen (133001)
- 2 Unterlegscheiben 6 x 2,2 x 0,5mm (132001)
- zwei Fußabdeckungen (131007, 131008)
- 8 Schrauben M2 x 4mm (132002)

Als nächstes verbinden Sie zwei der gerade verschraubten Bleche mit den zuvor verschraubten „Fuß-Servos“.
Dazu muss das U - Blech mit Gewinde HR1B-0001 leicht aufgebogen werden, damit es über die Verschraubung der Abtriebsräder des Servos rutschen kann
Achten Sie auf die Positionierung des Servos. Wenn das Blechteil im rechten Winkel zum Servo steht, muss beim linken Fuß (Servo Nr. 6) die auf dem Abtriebsrad eingeprägte „3“ auf „Position 12 Uhr“ stehen, beim rechten Fuß (Servo 8) muss die „7“ auf „Position 12 Uhr“ stehen. Siehe Bild unten!
Verschrauben Sie die Bauteile mit je 8 Schrauben 2mm x 4mm (132002).
Befestigen Sie außerdem das Servokabel, so wie auf dem Bild unten zu sehen. Nutzen Sie dazu die Kabelbefestigung (133001) und verschrauben sie diese durch das U-Blech in der unteren Position am Servogegenlager. Legen Sie unbedingt eine Unterlegscheibe zwischen Schraube und Kabelbefestigung.
Achten Sie darauf, dass die Servokabel beim bewegen der Servos nirgends eingeklemmt werden können.



Zum Schluss platzieren Sie die Fußabdeckungen wie im Bild zu sehen über die montierten Baugruppen (131007 - rechter Fuß, Servo Nr. 8, 131008 - linker Fuß, Servo Nr. 6, Prägung auf der Innenseite der Abdeckung) und verschrauben sie mit jeweils vier Schrauben M2x4mm. Die Fußabdeckungen sind zueinander symmetrisch. Eine Längsseite ist flach, eine Seite ist abgerundet. Die abgerundete Seite muss beim fertig montierten Roboter nach außen zeigen.



2.1.4 Schienbein:

Benötigte Bauteile:

- 4 I-Bleche HR1B-0003 (130002)
- ein Servo Nr.4 (134002)
- ein Servo Nr.2 (134003)

Das I-Blech HR1B-0003 (130002) hat produktionsbedingt zwei Seiten. Die eine Seite hat leicht abgerundete Kanten, die andere Seite ist relativ scharfkantig. Um einen sichereren Betrieb zu gewährleisten, sollte es immer so montiert werden, dass die leicht abgerundete Seite nach Außen (vom Servo weg zeigt).

Entfernen Sie beidseitig die 3 Schrauben aus dem 5-eckigen Teil der Servos (achten Sie auf die Positionen der unterschiedlich langen schwarzen und silbernen Schrauben) und befestigen Sie damit die I-Bleche wie oben beschrieben.
Am Ende dieses Abschnitts müssen auch hier wieder zwei symmetrische Baugruppen vor Ihnen liegen



2.1.5 Knie:

Benötigte Bauteile:

- 1x Servo Nr.3 Typ (134001, rechtes oberes Knie-Servo)
- 1x Servo Nr.2 (134002, rechtes unteres Knie-Servo)
- 1x Servo Nr.1 (134000 linkes oberes Knie-Servo)
- 1x Servo Nr.4 (134003 linkes unteres Knie-Servo)
- 2x H-Blech HR1B-0004 (130003)

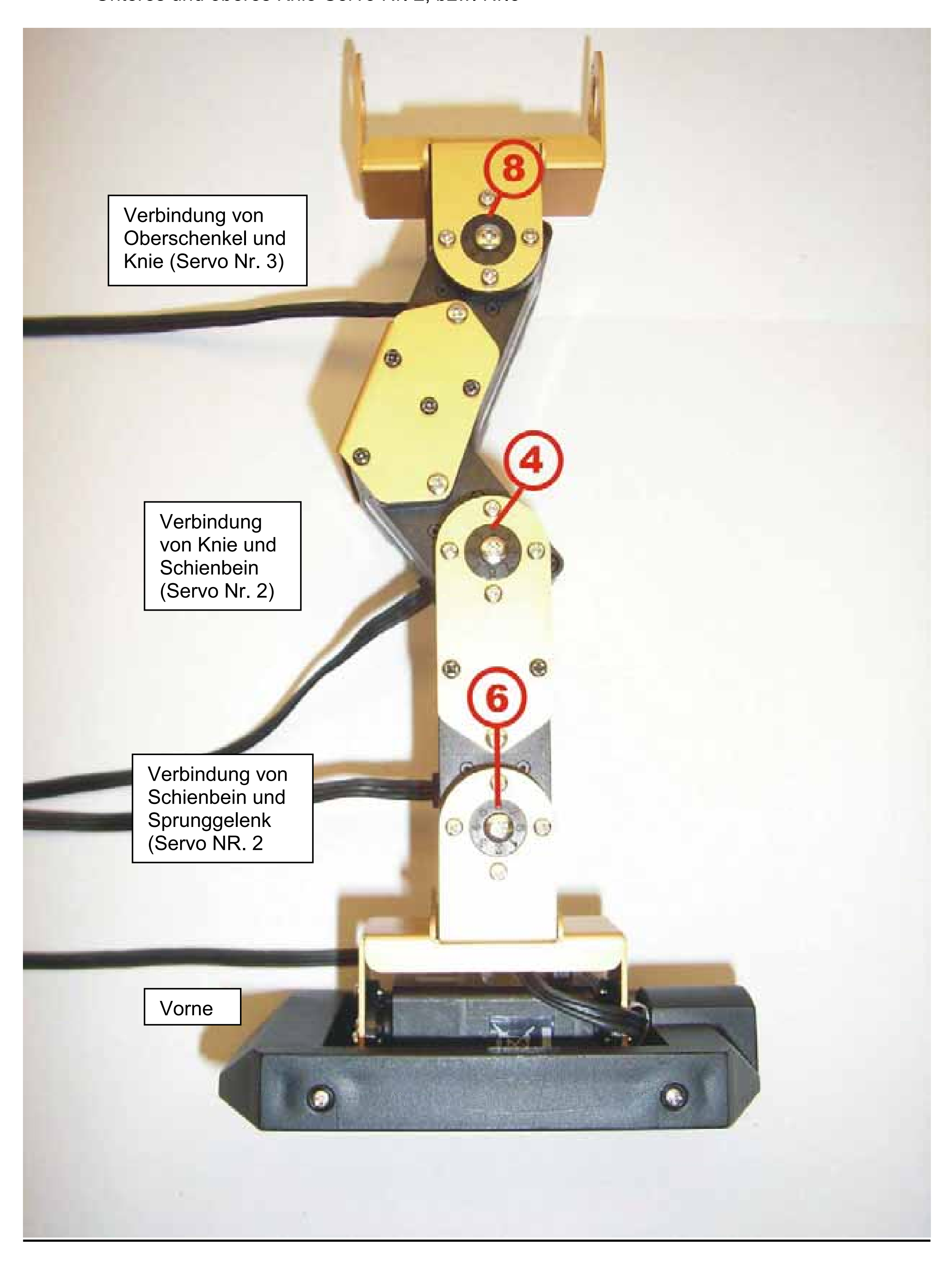
Entfernen Sie beidseitig die 3 Schrauben aus dem 5-eckigen Teil der 4 Servos (achten Sie auf die Positionen der unterschiedlich langen schwarzen und silbernen Schrauben).
Anschließend verbinden Sie das H-Blech HR1B-0004 (130003) und die entsprechenden oberen und unteren Servos mit den zuvor herausgedrehten Schrauben.
Nach der Montage sollten vor Ihnen wieder 2 symmetrische Baugruppen, wie auf dem Bild zu sehen, liegen. Die korrekte Servomontage können Sie anhand der Servoleitungen überprüfen: die Anschlüsse müssen jeweils nach außen zeigen, die kürzeren Leitungen befinden sich an den oberen Servos



2.1.6 *Montage des rechten Beins*

Es kommen die folgenden vormontierten Servos zum Einsatz:

- Fußgelenk-Servo Nr. 8
- Schienbein-Servo Nr. 2
- Unteres und oberes Knie-Servo Nr. 2, bzw. Nr.3



Verbindung von Fuß und Schienbein:

Befestigen Sie das Schienbein-Servo Nr. 2 so an dem U - Blech ohne Gewinde HR1B-0002 (130001) des rechten Fußes (Servo Nr. 8), dass das Servokabel nach vorne zeigt (siehe Bild oben). Dazu muss das Blech wieder leicht auseinander gebogen werden, damit es über die Schrauben gleiten kann
Achten Sie auf die im Abtriebsrad eingeprägten Nummern. Die „Position 6“ muss in 12-Uhr-Stellung stehen, wenn die Teile wie oben abgebildet positioniert sind.
Verschrauben sie die beiden Einheiten mit 8 silbernen Schrauben 2 x 4mm (132007).

Verbindung von Schienbein und Knie:

Befestigen Sie das I-Blech HR1B-0003 (130002) des „Schienbein-Servos“ Nr. 2 am Abtriebsrad des unteren „Knie-Servos“ Nr. 2. Auch hier zeigen die Servokabel nach vorne.
Achten Sie auf die im Abtriebsrad eingeprägten Nummern. Die „Position 4“ muss in 12-Uhr-Stellung stehen, wenn die Teile wie oben abgebildet positioniert sind.
Verschrauben sie die beiden Einheiten mit 8 silbernen Schrauben 2 x 4mm (132007).

Verbindung von Knie und Oberschenkel:

Befestigen Sie das in Schritt *2.1.2.* montierte „Hüft-Teil“ mit dem U - Blech ohne Gewinde (die Schraubenköpfe zeigen zum Servo) am „oberen Knie-Servo“ Nr. 3.
Achten Sie auf die im Abtriebsrad eingeprägten Nummern. Die „Position 8“ muss in 12-Uhr-Stellung stehen, wenn die Teile wie oben abgebildet positioniert sind.
Verschrauben sie die beiden Einheiten mit 8 silbernen Schrauben 2 x 4mm (132007).

Achtung:
Die Beine müssen korrekt montiert werden, damit die Servos anschließend den vollen Bewegungsbereich ausnutzen können!

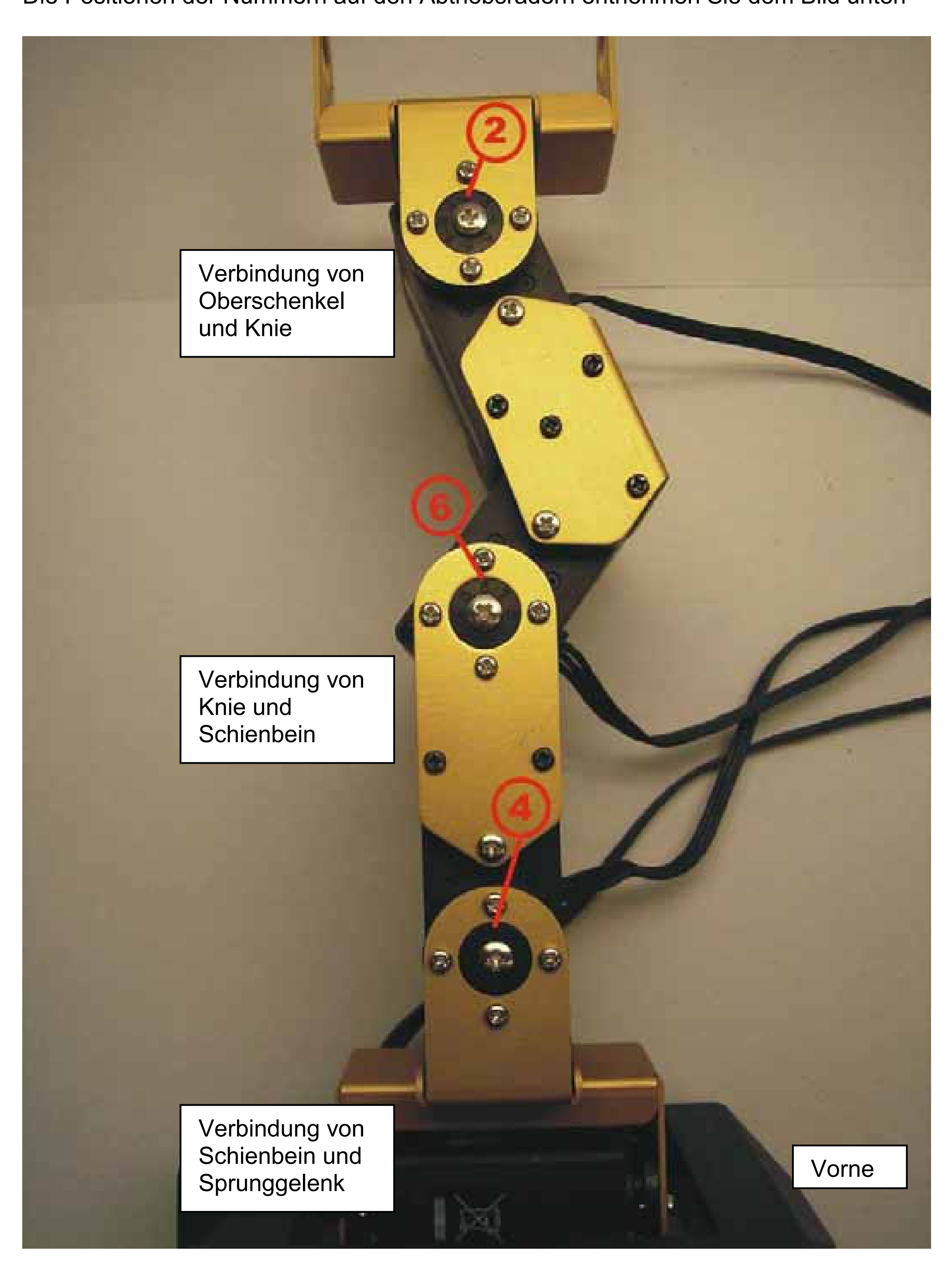
2.1.7 *Montage des linken Beins*

Die Montage des linken Beins erfolgt auf die selbe Weise wie oben beschrieben und unterscheidet sich lediglich bei den Nummern der Servos, sowie den eingeprägten Positionen auf den Abtriebsrädern.

Es kommen die folgenden vormontierten Servos zum Einsatz:

- Fußgelenk-Servo Nr. 6
- Schienbein-Servo Nr. 4
- Oberes und unteres Knie-Servo Nr. 1 und Nr. 4

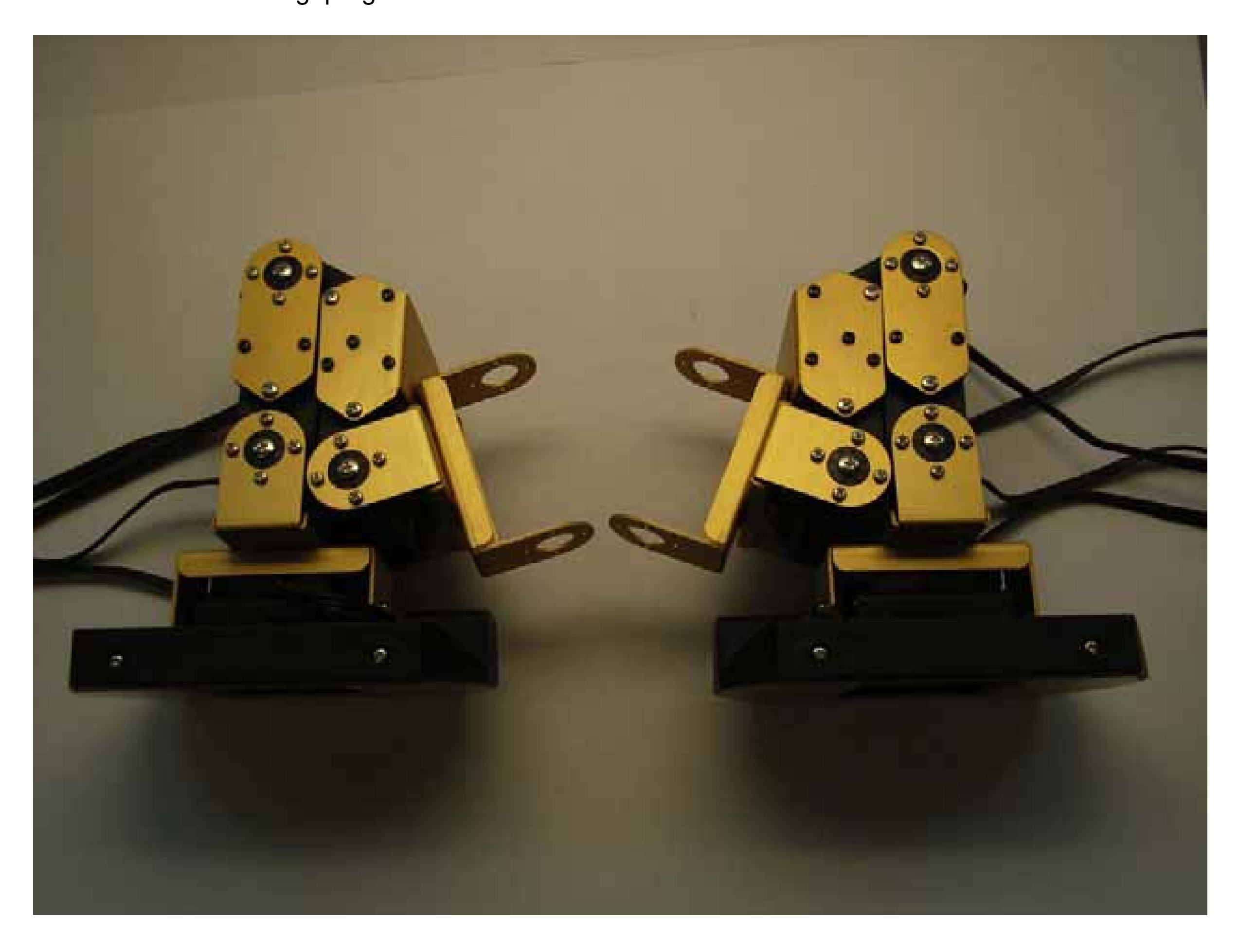
Die Positionen der Nummern auf den Abtriebsrädern entnehmen Sie dem Bild unten



Die fertig montierten Beine müssen so aussehen wie auf dem Bild unten zu sehen:



Wenn Sie die Beine nicht wie auf dem Bild gezeigt zusammenfalten können, dann kontrollieren Sie bitte die vorausgegangen Montageschritte, besonders die Positionen der in den Abtriebsrädern eingeprägten Nummern.



2.1.8 Aufbau der Arme

2.1.9 Oberarm

Benötigte Bauteile:

- 4 I-Bleche HR1B-0003 (130002)
- 1x Servo Nr. 2 (134000)
- 1x Servo Nr. 4 (134001)

Das I-Blech HR1B-0003 (130002) hat produktionsbedingt zwei Seiten. Die eine Seite hat leicht abgerundete Kanten, die andere Seite ist relativ scharfkantig. Um einen sichereren Betrieb zu gewährleisten, sollte es immer so montiert werden, dass die leicht abgerundete Seite nach Außen zeigt.

Entfernen Sie beidseitig die 3 Schrauben aus dem 5-eckigen Teil der Servos und befestigen Sie damit die I-Bleche wie oben beschrieben. Achten sie darauf, dass die, in ihrer Länge unterschiedlichen, silbernen und schwarzen Schrauben jeweils wieder in der Position eingeschraubt werden, aus der sie vorher entfernt wurden.
Am Ende dieses Abschnitts müssen auch hier wieder zwei symmetrische Baugruppen vor Ihnen liegen



2.1.10 *Unterarm*

Benötigte Bauteile:

- 1x Servo Nr.11 (links)
- 1x Servo Nr.12 (rechts)
- 2 Hand-Teile HR1C-0006 (131006)
- 2 Hand-Teile HR1C-0007 (131007)
- 2 Handverbindungsbleche HR1B-00011 (130010)
- 8 Schrauben, 2x5mm (132008)

Entfernen Sie an jedem Servo die 4 Schrauben, die vom Abtriebsrad, bzw. dem Gegenlager abgewandt sind und befestigen Sie damit die Handverbindungsbleche (HR1B-00011) wie im Bild unten zu sehen.
Für die Hand stecken sie jeweils ein Teil HR1C-0006 und HR1C-0007 zusammen und verschrauben es mit jeweils vier Schrauben, 2x5mm (132008) am Handverbindungsblech.
Die Servokabel müssen nach außen zeigen!

Wieder müssen zwei symmetrische Baugruppen vor Ihnen liegen.



2.1.11 *Montage der ganzen Arme*

Benötigte Bauteile:

Rechter Arm:
- Oberarm (Servo Nr. 4)
- Unterarm (Servo Nr.12)
- 8 Schrauben 2x4mm (132007)

Linker Arm:
- Oberarm (Servo Nr. 2)
- Unterarm (Servo Nr.11)
- 8 Schrauben 2x4mm (132007)

Verbinden Sie die zuvor montierten Oberarm- und Unterarm-Baugruppen. Verwenden Sie dazu pro Arm 8 Schrauben 2x4mm (132007)
Die Leitungen der Oberarm- und Unterarm-Servos müssen dabei nach Außen zeigen (siehe Bild unten).

Achten Sie auf die im Abtriebsrad eingeprägten Nummern. Die „Position 1“ muss jeweils in 12-Uhr-Stellung stehen, wenn die Teile wie unten abgebildet positioniert sind.

2.2 Aufbau des Körpers

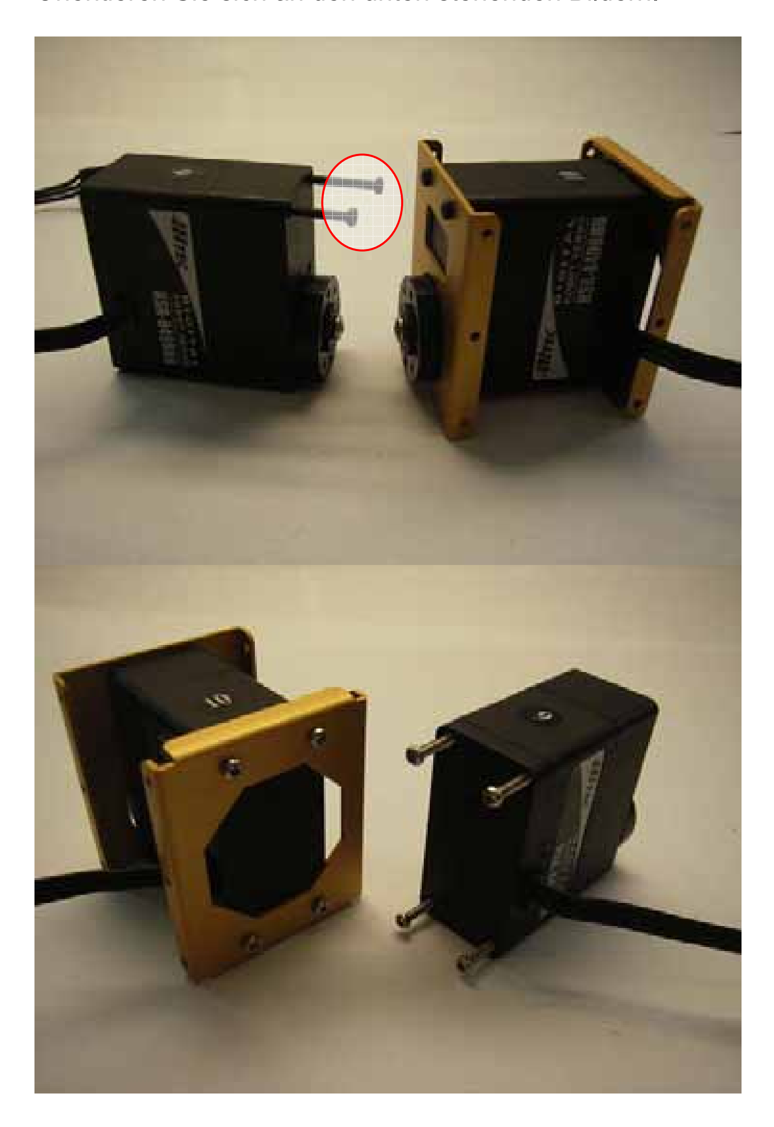
2.2.1 Schulterservos

Es werden die folgenden Teile benötigt:

- 1x Servo Nr. 9 (134008, linke Schulter)
- 1x Servo Nr. 10 (134009, rechte Schulter)
- 2 Schulter-Innenbleche HR1B-0009 (130008)
- 2 Schulter-Außenblech HR1B-0010 (130009)

Entfernen Sie die zwei, vom Abtriebsrad abgewandten, schwarzen Schrauben aus den Servos und befestigen damit das Schulter-Außenblech HR1B-0010 (130009) an jedem Servo. Anschließend entfernen Sie die jeweils vier silbernen Schrauben aus der Unterseite der Servos und befestigen damit das Schulter-Innenblech HR1B-009 (130008) an jedem Servo.

Orientieren Sie sich an den unten stehenden Bildern.



2.2.2 Montage des Körper-Vorderteils:

Benötigt werden die folgenden Teile:
- 1x Servo Nr. 5 (134004)
- 1x Servo Nr. 7 (134005)
- Körper Vorderteil HR1B-0006 (130005)
- Die beiden zuvor montierten Schulterelemente mit den Servos Nr.9 und Nr.10 (*Schritt 2.2.1*)
- 8 Schrauben M2x4mm (132002)

Entfernen Sie die Abtriebsräder von den beiden Servos.
Befestigen Sie die Servos am Körper Vorderteil HR1B-0006 (130005), dass die Leitungen, wie im Bild zu sehen, nach innen zeigen. Entfernen sie an den Servos dazu jeweils die zwei nach außen zeigenden schwarzen Schrauben aus der Oberseite und verschrauben diese jeweils wieder durch das Körper Vorderteil HR1B-0006.
Anschließend müssen die beiden Abtriebsräder wieder montiert werden.
Montieren Sie die beiden Schulter-Elemente mit jeweils 4 Schrauben M2x4mm (132002) an das Körper Vorderteil HR1B-0006.
Die Servoleitungen müssen jeweils zur Rückseite des Körpers zeigen (siehe Bild)!

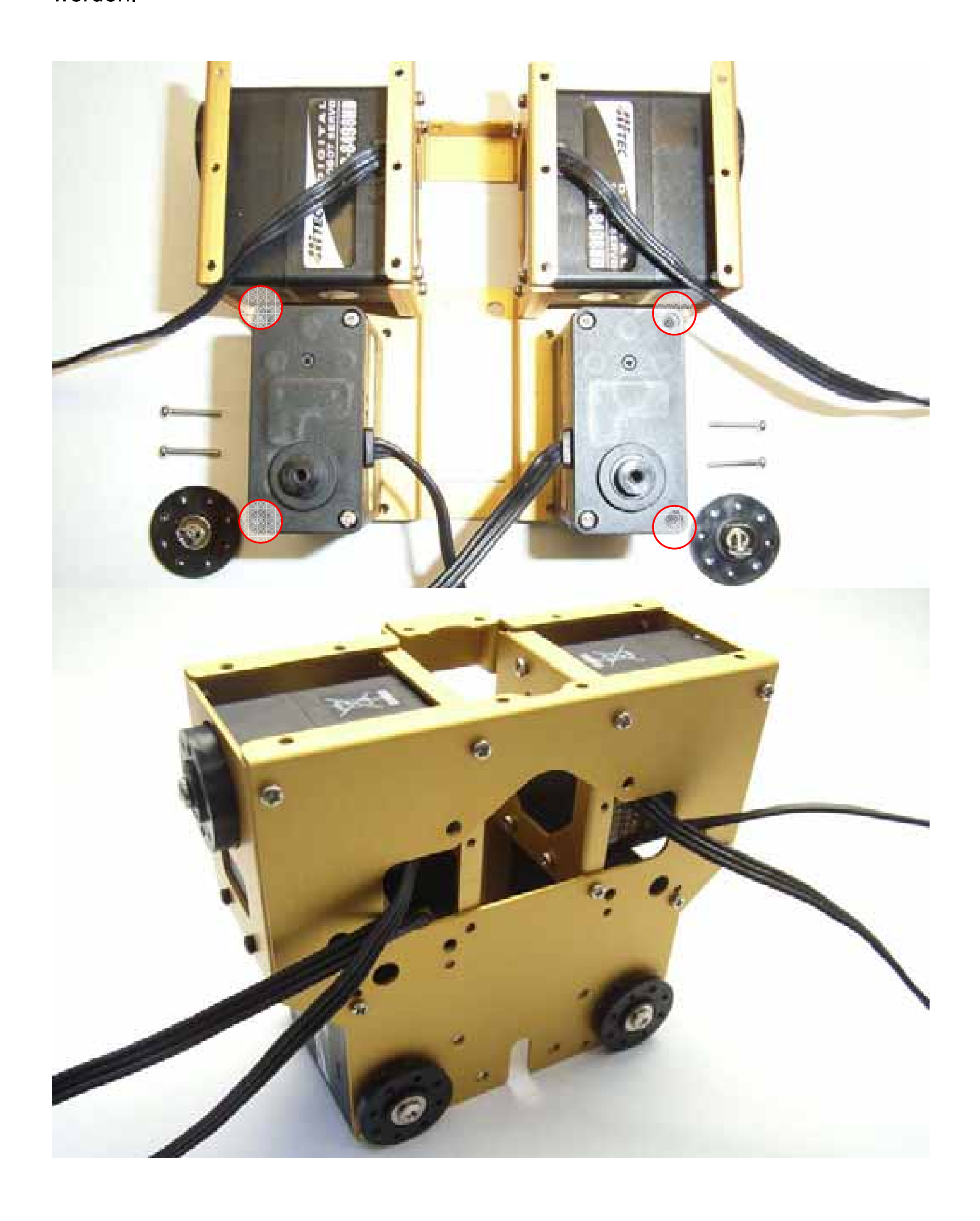


2.2.3 Montage der Körper Rückseite HR1B-0005

Benötigt werden die folgenden Teile:

- Vormontiertes Körpervorderteil (*Schritt 2.3.2*)
- Körper Rückseite HR1B-0005 (130004)
- 8 Schrauben M2x4mm (132002)

Entfernen Sie die hinteren Servo-Gegenlager von den Servos Nr. 5 und Nr. 7 im Vormontierten Körpervorderteil.
Entfernen Sie am Servo Nr. 5 die zwei linken Schrauben an der Außenseite und am Servo Nr. 7 die zwei rechten Schrauben an der Außenseite.
Führen sie die Servokabel durch das große Loch in der Körper Rückseite HR1B-0005 (130004) und platzieren Sie diese auf dem Vormontierten Körper-Vorderteil. Verschrauben Sie die beiden Teile mit 8 Schrauben M2x4mm (132002) an den Schulterteilen und mit den zuvor aus den Servos Nr. 5 und Nr. 7 entfernten Schrauben an den Servos.
Nun können die hinteren Servo-Gegenlager wieder an den Servos Nr. 5 und Nr. 7 montiert werden.

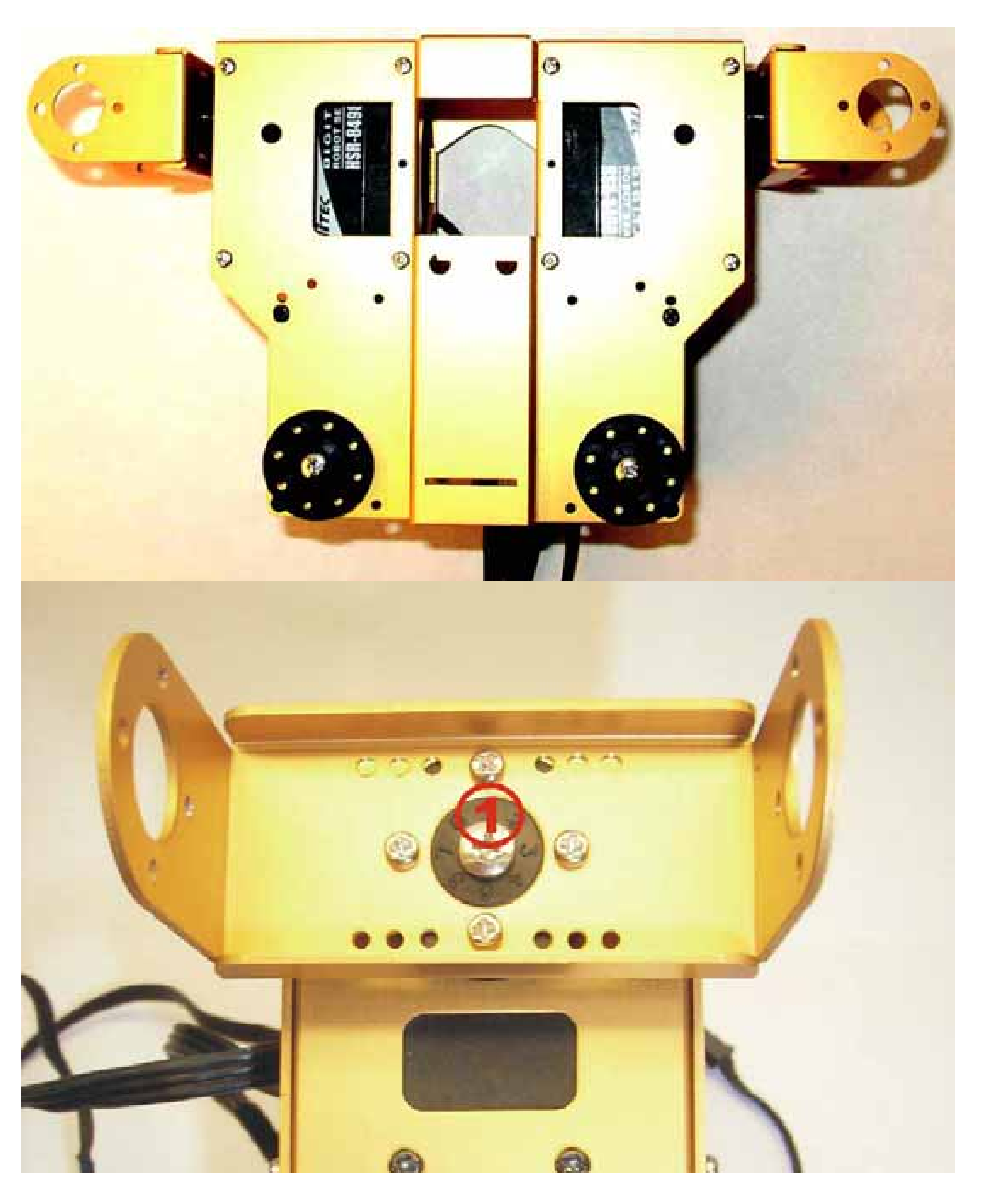


2.2.4 *Schulter-Oberarm-Verbindung*

Benötigte Bauteile:

- den in den Schritten zuvor montierten Körper
- 2 U-Bleche ohne Gewinde HR1B-0002 (130001)
- 8 Schrauben 2x4mm (132007)

Montieren Sie jeweils ein U-Blech ohne Gewinde HR1B-0002 (130001) an die Abtriebsräder der Schulter-Servos Nr. 9 und Nr. 10. Verwenden Sie hierzu jeweils 4 Schrauben 2x4mm (132007).
Achten Sie auf die im Abtriebsrad eingeprägten Nummern. Die „Position 1“ muss in 12-Uhr-Stellung stehen, wenn die Teile wie unten abgebildet positioniert sind.

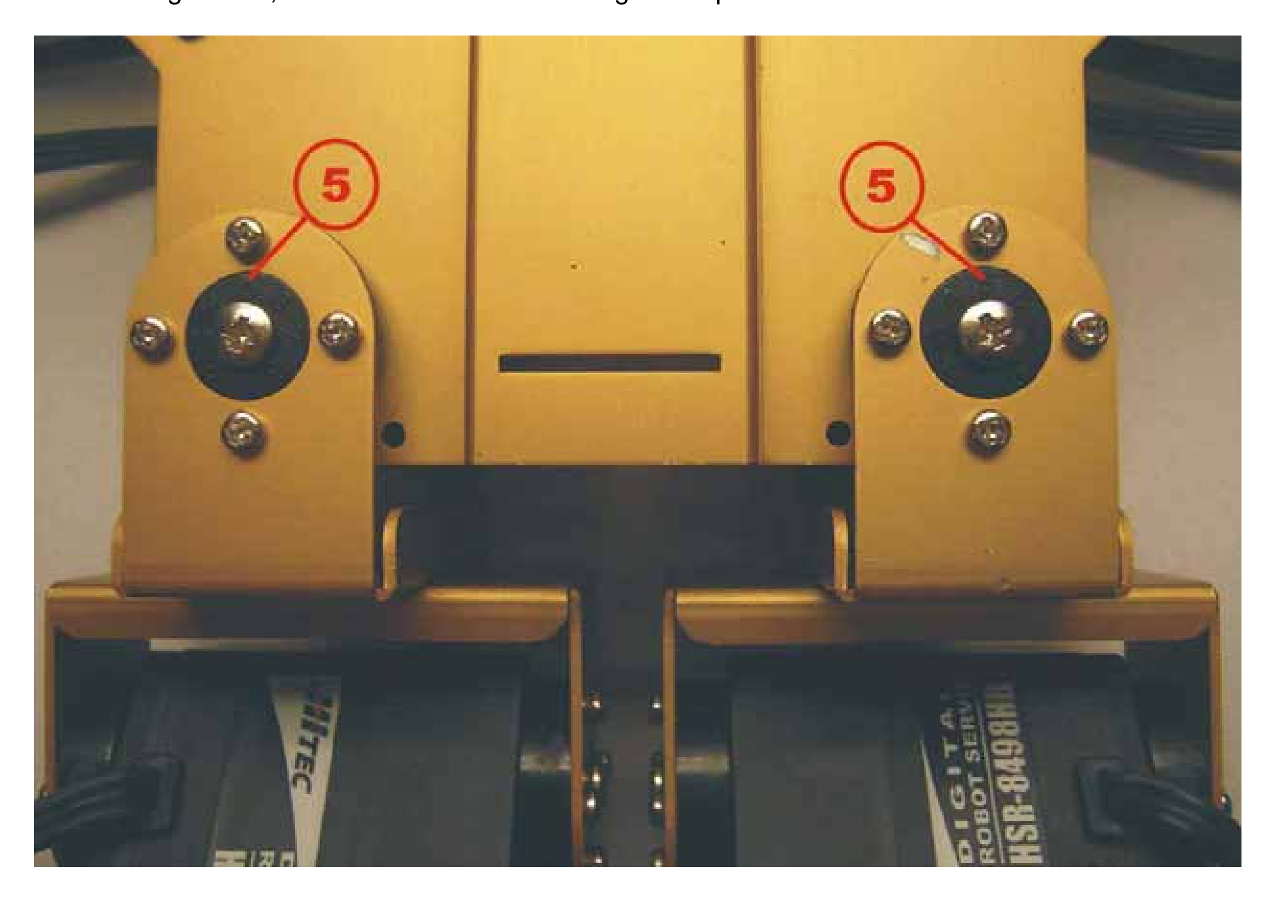


2.3 Zusammenfügen von Beinen, Armen, Kopf und Körper

2.3.1 Montage von Beinen und Körper

Montieren Sie die vormontierten Baugruppen für Beine und Körper so wie auf dem Bild unten zu sehen zusammen.
Verwenden Sie dazu für jedes Servo je 8 Schrauben 2x4mm (132007)

Achten Sie auf die im Abtriebsrad eingeprägten Nummern. Die „Position 5" muss jeweils in 12-Uhr-Stellung stehen, wenn die Teile wie unten abgebildet positioniert sind.

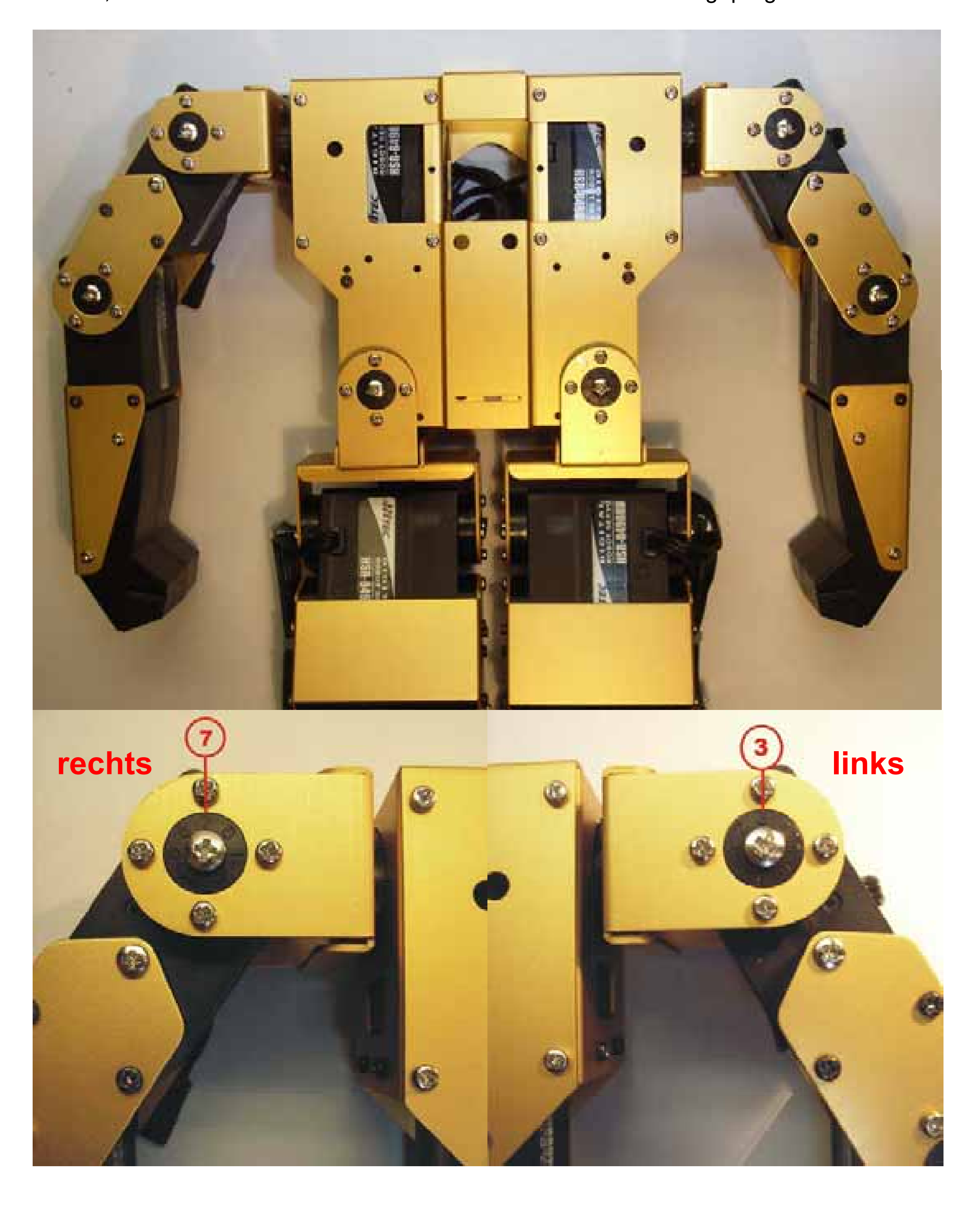


2.3.2 *Montage von Armen und Körper*

Montieren Sie die vormontierten Baugruppen für Arme und Körper so wie auf dem Bild unten zu sehen zusammen. Verwenden Sie dazu für jedes Servo je 8 Schrauben 2x4mm (132007)

Achten Sie auf die im Abtriebsrad eingeprägten Nummern. Beim rechten Arm (Servos Nr. 4 und Nr. 12) muss die „Position 7" in 12-Uhr-Stellung stehen, beim linken Arm (Servos Nr. 2 und Nr. 11) muss die „Position 3" in 12-Uhr-Stellung stehen wenn die Teile wie unten abgebildet positioniert sind.

Überprüfen Sie nach der Montage bitte, ob sich die Arme so wie vorgesehen bewegen können. Beide Arme müssen sowohl waagrecht nach vorne und hinten, als auch senkrecht nach oben und unten schwenken können.
Sollte dies nicht der Fall sein, dann überprüfen sie bitte die korrekte Montage des betroffenen Servos, insbesondere die Positionen der in den Abtriebsrädern eingeprägten Nummern.



2.3.3 *Montage von Kopf und Körper*

Es werden die folgenden Teile benötigt:

- Servogegenlager HSR8498HA2 (135004)
- Körperoberteil HR1B-0007 (130006)
- LED Platine (136001)
- Visier HR1C-0003 (131003)
- Kopf Vorderteil HR1C-0004 (131003)
- Kopf Hinterteil HR1C-0005 (131004)
- 2 Schrauben 2x5mm (132008)
- 2 Schrauben 2x4mm (132007)
- 10 Schrauben M2.6x4 (132003)
- 6 Schrauben 2x8mm (13209)

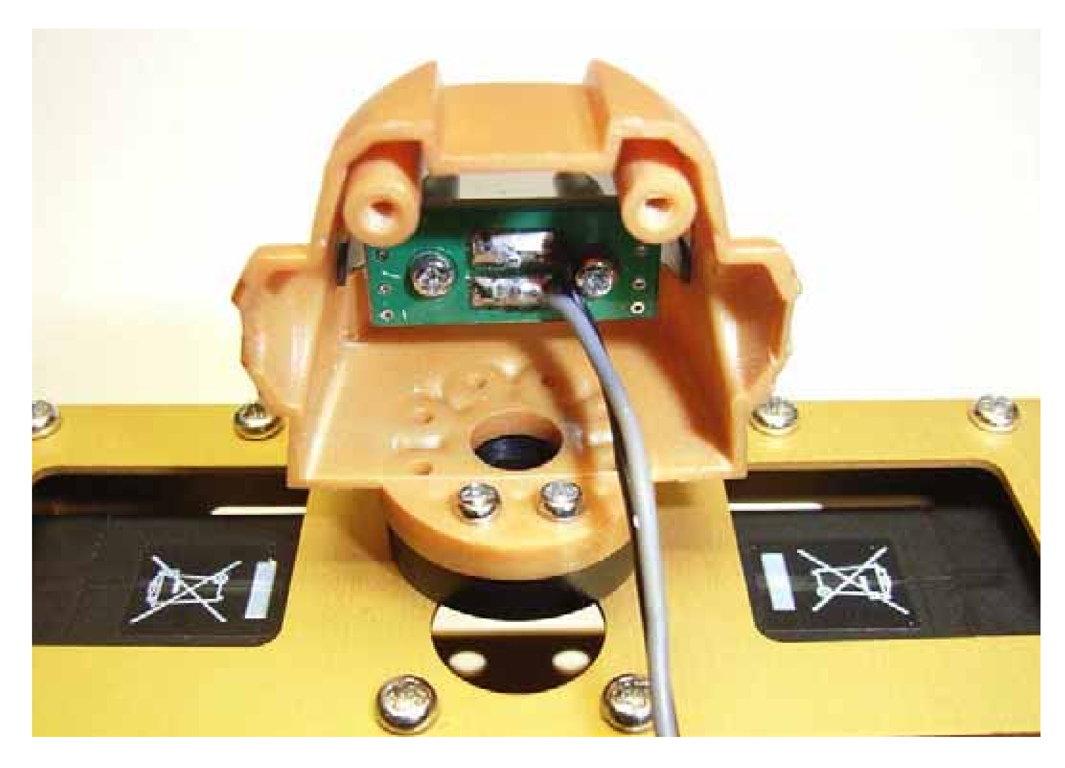
Schrauben Sie das Servogegenlager HSR8498HA2 (135004) mit vier Schrauben 2x8mm (132008) an das Körperoberteil HR1B-0007 (130006).
Schrauben sie das Körperoberteil HR1B-0007 mit 10 Schrauben M2.6x4 (132003) auf den, in den Schritten zuvor montierten, Körper.



Schrauben sie die LED Platine (136001) mit 2 Schrauben 2x4mm (132007) in das Visier HR1C-0003 (131003).



Schieben Sie das Visier mit der LED Platine von vorne in das Kopf Vorderteil HR1C-0004 (131004), bis die Haltehaken im Kopf einrasten.
Montieren Sie das Kopf Vorderteil HR1C-0004 (131004) mit zwei Schrauben 2x8mm (13209) auf das Servogegenlager (wie im Bild unten).
Danach schrauben Sie das Kopf Hinterteil HR1C-0005 (131005) an das Vorderteil. Benutzen Sie dazu zwei Schrauben 2x5mm (132008).



2.3.4 Montage des Remocon-IR-Sensors

Befestigen Sie den Remocon-IR-Sensor mit einem Stück doppelseitigen Klebeband auf dem zuvor montierten Kopf



2.3.5 Montage der vorderen Abdeckung

Benötigt werden:

- Abdeckung vorne HR1C-0001 (131000)
- 2 Schrauben 2x5mm (132008)

Legen Sie die vordere Abdeckung HR1C-0001 (131000) auf den Körper und verschrauben sie diese mit zwei Schrauben 2x5mm (132008) von hinten durch die Rückwand.



Dieser Schritt gelingt am einfachsten mit einem magnetischen Schraubendreher. Alternativ halten sie den Roboter mit der Rückseite nach unten und führen den Schraubendreher mit der Schraube von unten durch das Loch.

2.4 Controller und Kabel

2.4.1 Einbau des MR-C3024 Controller

Benötigt werden:

- 4 Abstandsbolzen mit Gewinde M3 (132005)
- 4 Schrauben M3x4mm (132004)
- ein MR-C3024 Controller

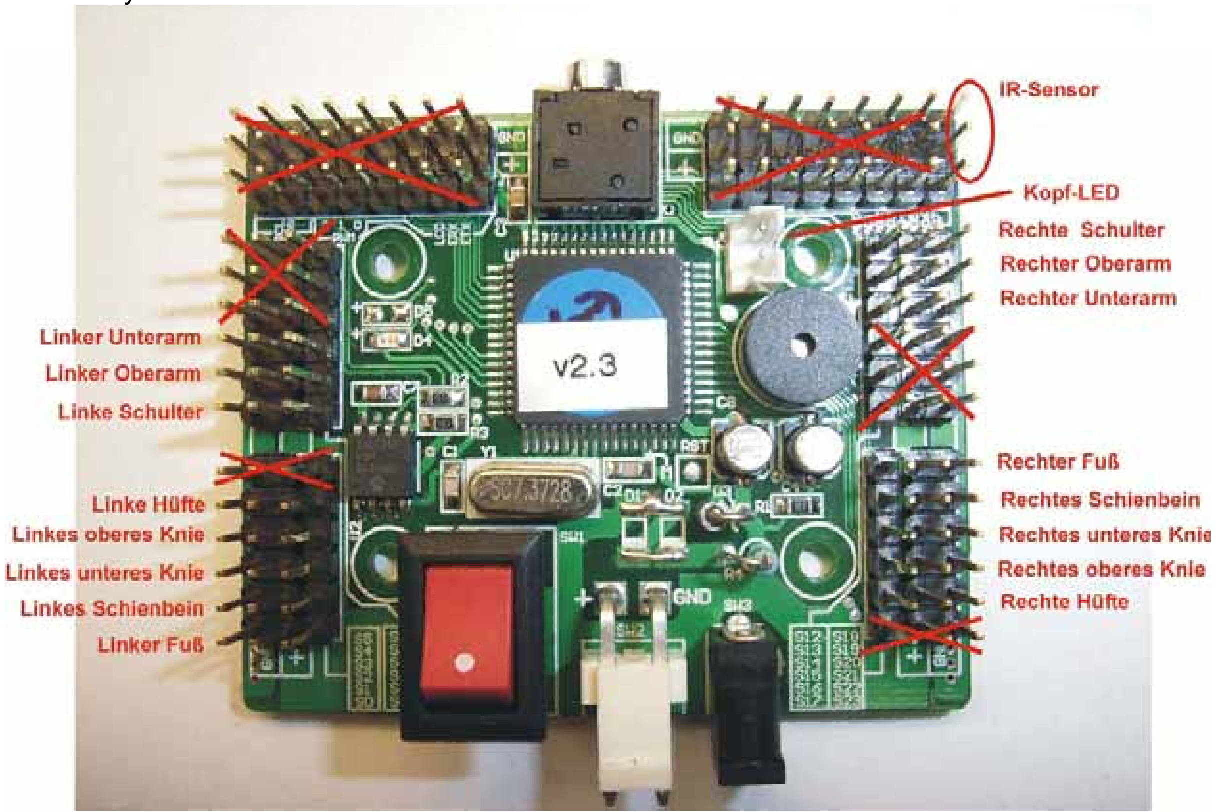
Die vier Abstandsbolzen mit Gewinde M3 (132005) werden in die dafür vorgesehenen Gewindelöcher in der Körper Rückseite HR1B-0005 (130004) eingeschraubt.
Verschrauben Sie den MR-C3024 Controller mit vier Schrauben M3x4mm (132004) an den Abstandsbolzen.



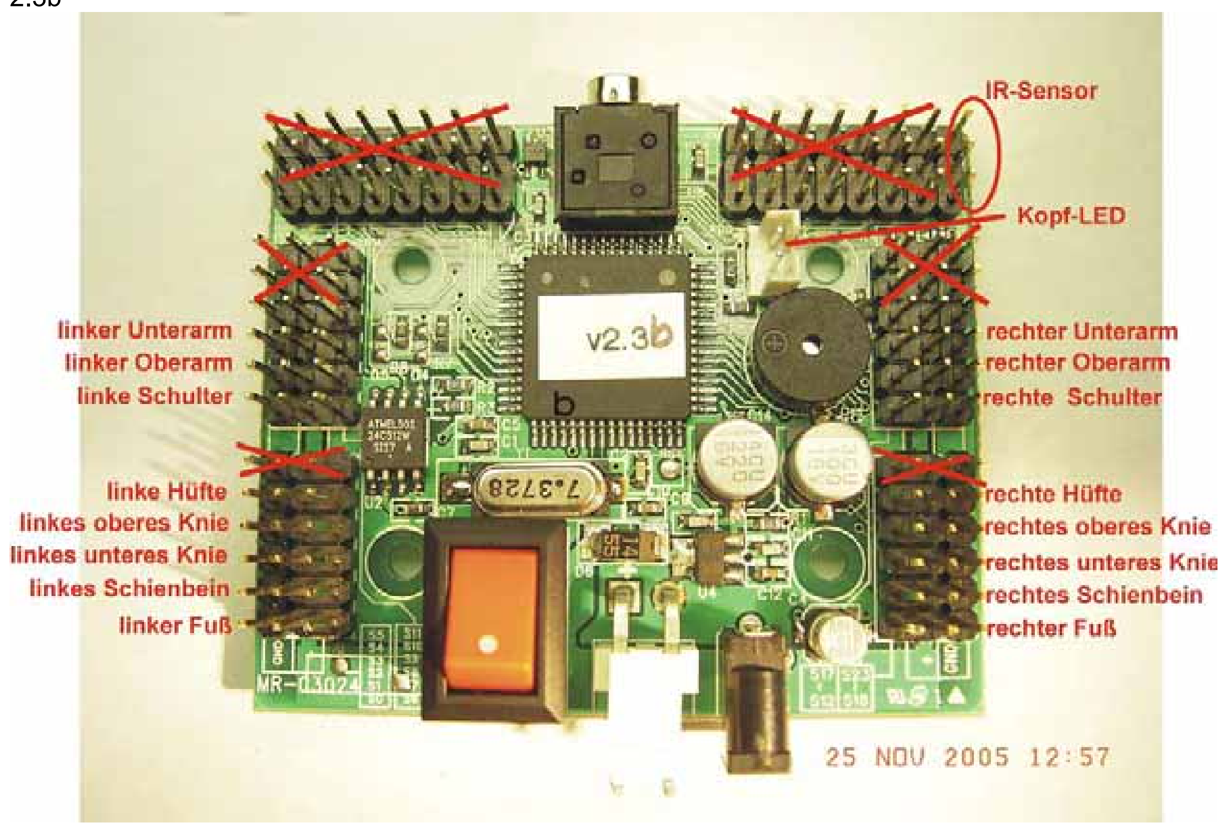
2.4.2 Anschluss der Servokabel

Stecken sie die Servoleitungen in die auf dem Bild angegebenen Positionen. Achten Sie dabei auf die Codierungsnasen an den Steckern. Diese müssen immer zur Innenseite zeigen. Stecken Sie außerdem den Stecker der LED-Platine und den IR-Sensor in die dafür vorgesehene Buchse (die Codierung muss ebenfalls nach innen zeigen).

Platine Layout 2.3



Platine Layout 2.3b



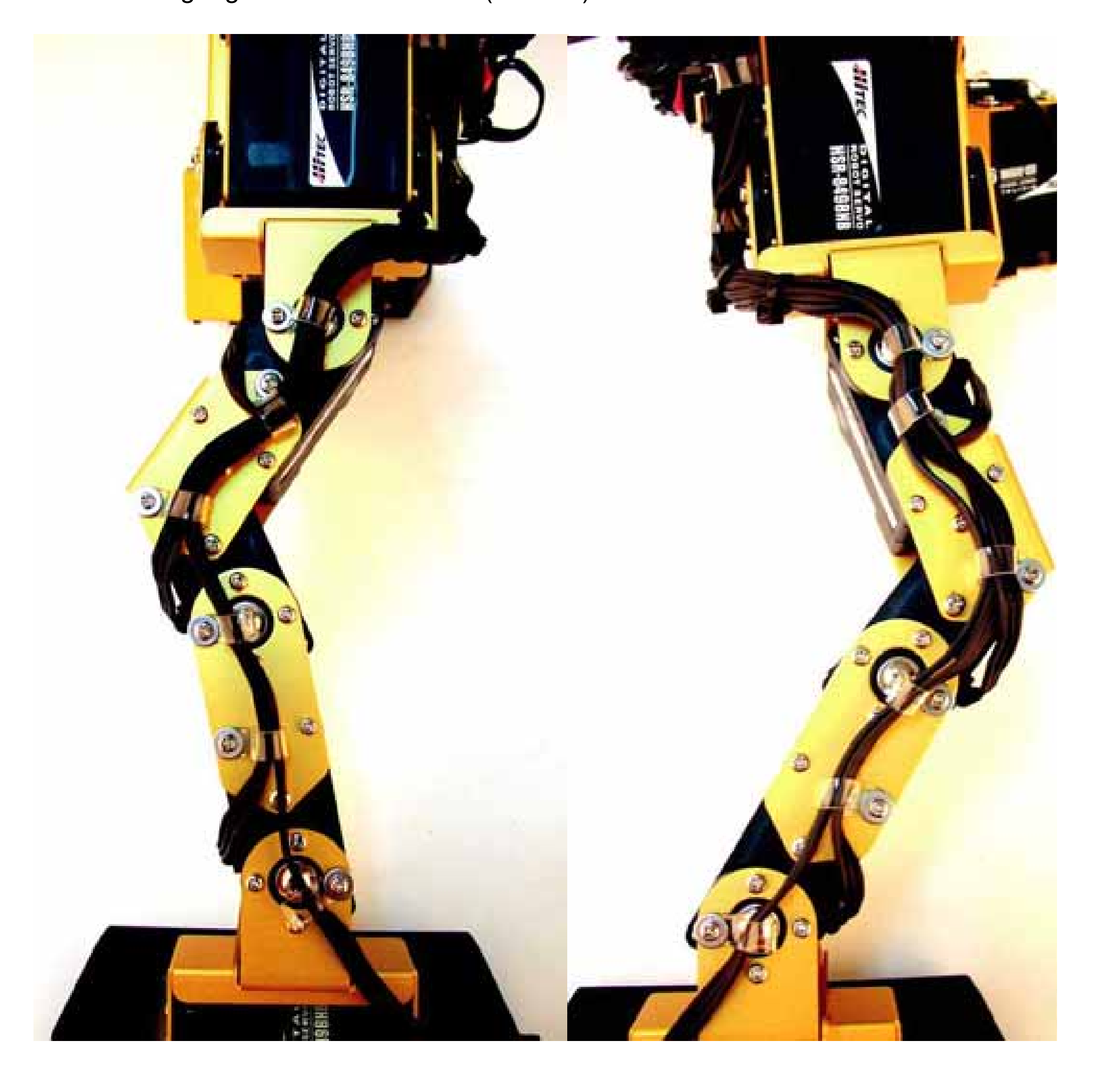
2.4.3 *Leitungsverlegung*

Benötigt werden:

- 28 Kabelbefestigungen (133001)
- 28 Unterlegscheiben 6 x 2,2 x 0,5mm (132001)
- 2 Kabelbefestigungen (133003)
- 8 Kabelbinder (133000)

Verlegen Sie die Servoleitungen so, dass sie unter keinen Umständen an Kanten und Ecken scheuern oder eingequetscht werden können! Außerdem ist darauf zu achten, dass die Leitungen dem vollen Bewegungsumfang der Servos folgen können und keine Zugbelastungen auf die Kabel entstehen!

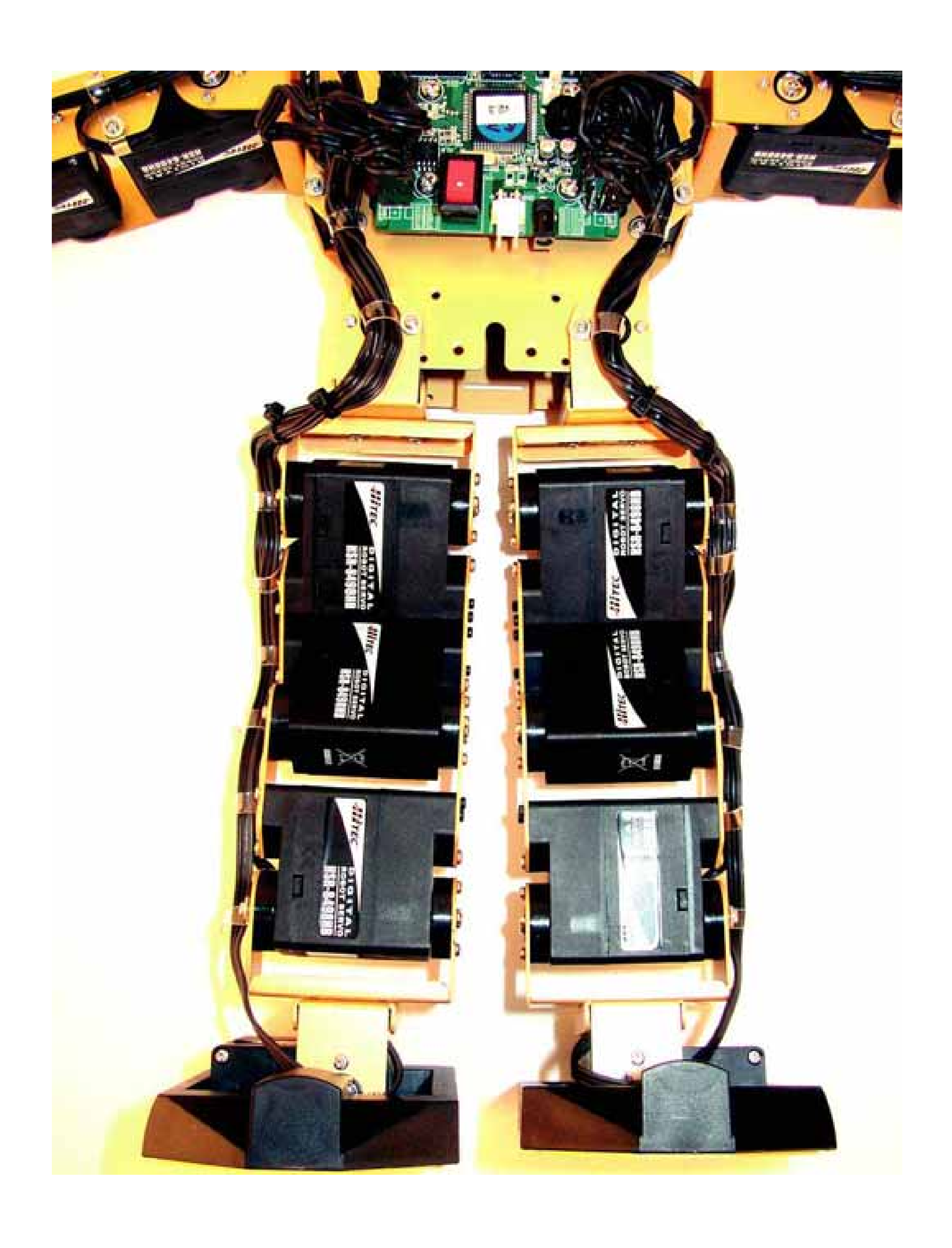
Orientieren Sie sich an den Bildern unten.
Entfernen Sie eine Schraube aus den Servogegenlagern auf der Rückseite des Roboters und befestigen Sie damit die Kabelbefestigungen (133001) am Roboter. Verwenden Sie zwischen Schraube und Kabelbefestigung jeweils eine Unterlegscheibe (132001).
Sichern und fixieren Sie die Kabelstränge an Armen und Beinen zwischen den Kabelbefestigungen mit Kabelbindern (133000).





Verwenden Sie die Kabelbefestigung (133003) so wie auf den Bildern unten zu sehen





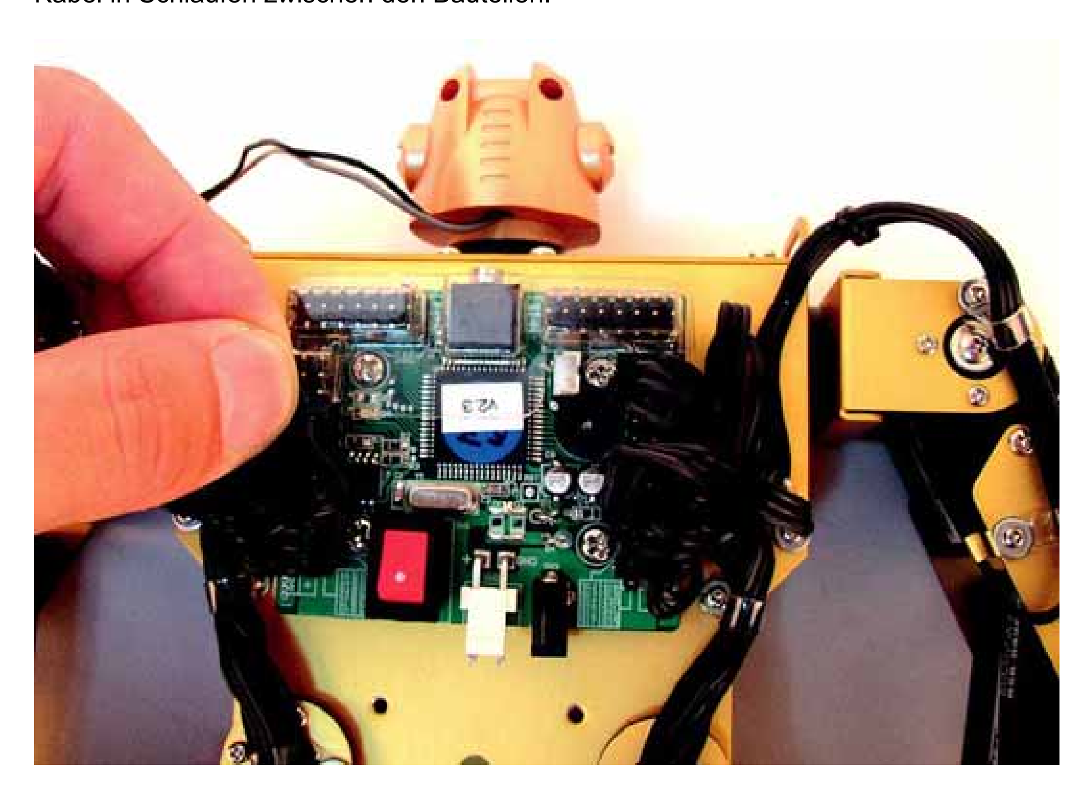
2.5 Endmontage

2.5.1 Montage der hinteren Abdeckung

Benötigt werden:

- Abdeckung hinten HR1C-0002 (131001)
- Pin-Abdeckung (131010)
- 2 Schrauben M2,6x4mm (132003)
- 2 Schrauben 2x26mm (132013)

Platzieren Sie die durchsichtige Pin-Abdeckung (131010) auf dem Controller Board wie auf dem Bild zu sehen.
Legen Sie die hintere Abdeckung auf die Rückseite des Roboters und verschrauben Sie unten mit zwei Schrauben M2, 6x4 (132003) und oben mit zwei Schrauben 2x26mm (132013).
Achten sie darauf, dass keine Servokabel eingeklemmt und gequetscht werden!
Schieben Sie die längeren Servoleitungen unter den Controller und verlegen Sie die restlichen Kabel in Schlaufen zwischen den Bauteilen.



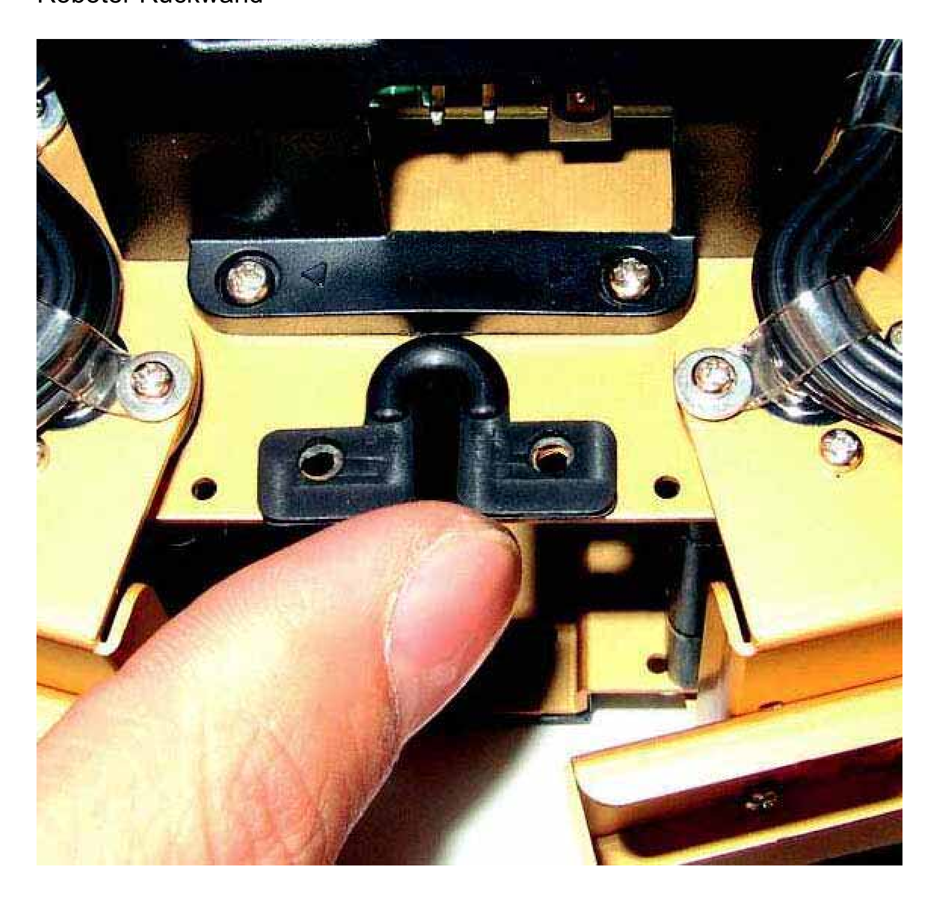

HITEC
POWER

2.5.2 *Einsetzen des Akkus*

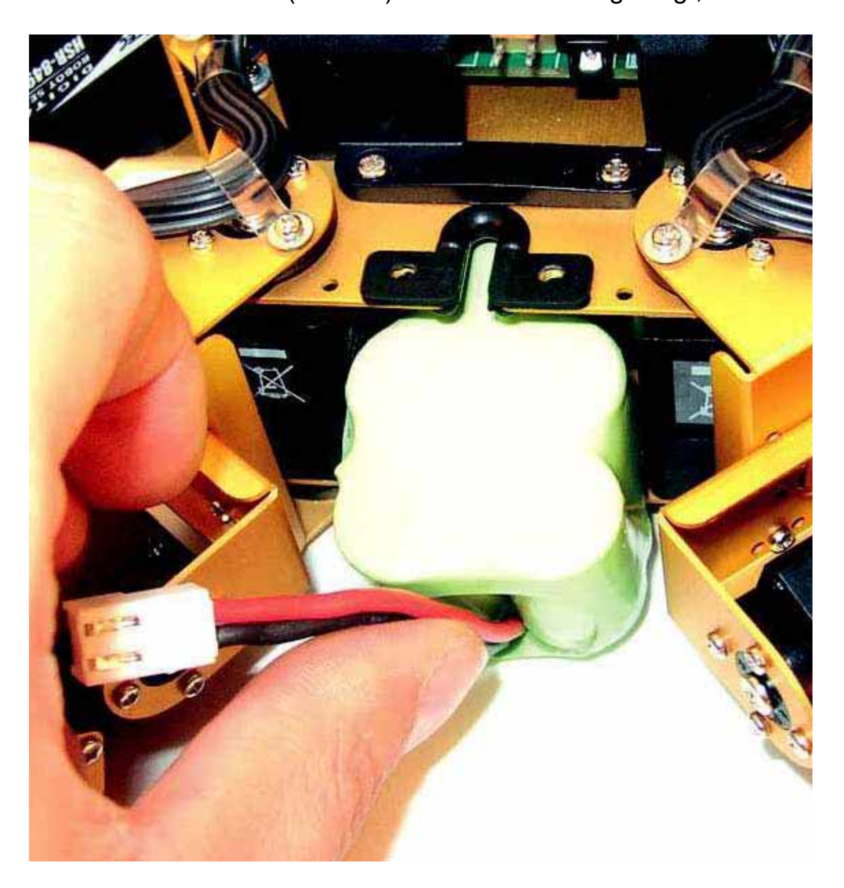
Benötigt werden:

- Akkukabelschutz (133002)
- 2 Rändelschrauben M3 (132006)
- Akku 6V / 1000mAh (136010)
- Körper Unterteil HR1B-0008 (130007)

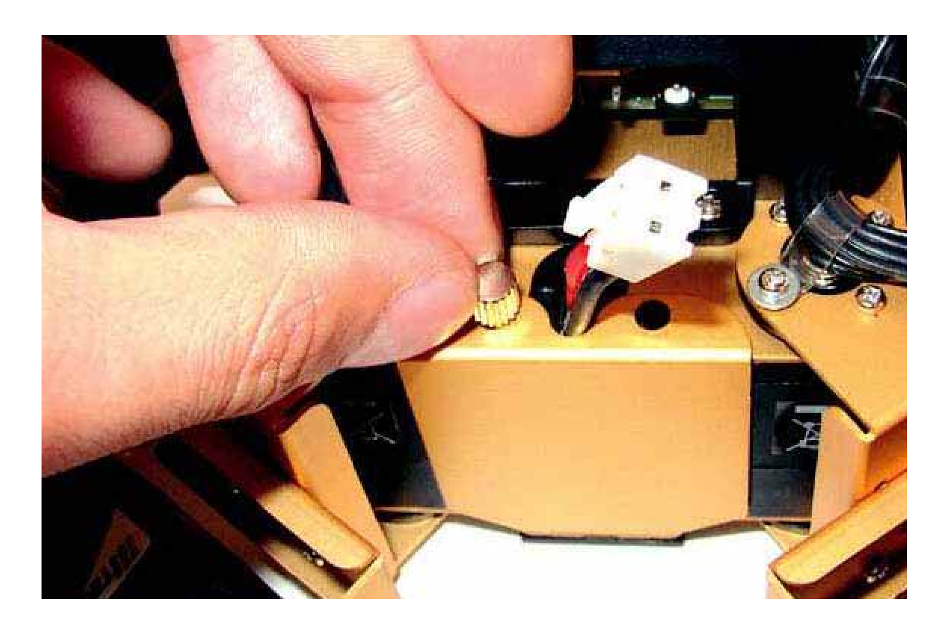
Schieben Sie zuerst der Akkukabel Schutz in den dafür vorgesehenen Ausschnitt in der Roboter Rückwand



Legen Sie den Akku 6V / 1000mAh (136010) wie im Bild unten gezeigt, in den Roboter ein.



Hängen Sie das Körper Unterteil HR1B-0008 (130007) mit der Lasche in das Körper-Vorderteil ein und befestigen es mit den zwei Rändelschrauben M3 (132006).
Stecken Sie den Akkustecker in die Akkubuchse auf der Controller Platine.





ROBONOVA-I

Installation und Betrieb von RoboBASIC, RoboScript, RoboRemocon

Inhaltsverzeichnis

Eingetragene Warenzeichen
Windows ist ein eingetragenes Warenzeichen der Microsoft Corporation

Information
Dies ist die Installationsanleitung für RoboBASIC v2,5, RoboScript v2.5, RoboScript v2.5 kann von früheren Versionen abweichen. An Software und Handbuch können zur Verbesserung ohneBekanntgabe Änderungen vorgenommen werden.

RoboBasic ist eine registrierte Software. Erstellen von Kopien, Veröffentlichung, Versand oder Vertrieb dieser Anleitung oder der Software ohne Genehmigung ist untersagt.

Hinweise zu RoboBASIC
RoboBASIC baut auf der Programmiersprache BASIC auf und ist für den Anschluss an die MR-C Robot Controller Produktreihe vorgesehen. RoboBASIC ist Entwicklungsumgebung für die Hitec/Multiplex Roboter Controller Serie. RoboBASIC ist unter Microsoft Windows XP lauffähig.

Installationsanleitung RoboBASIC

RoboBASIC Software kann von der dem ROBONOVA Bausatz beiliegenden CD installiert werden. Die aktuellste Version befindet sich auf HITEC Robotik Homepage www.robonova.de. Wir empfehlen, die aktuellste Version von Internet einzusetzen.

Hinweis: Seit der Softwareversion 2.5 werden die Programme RoboBASIC, RoboSCRIPT und RoboREMOCON in einem Installationspaket angeboten und gleichzeitig installiert.

- Starten Sie den Installationsdialog durch Klick auf das "setup.exe" Programm.

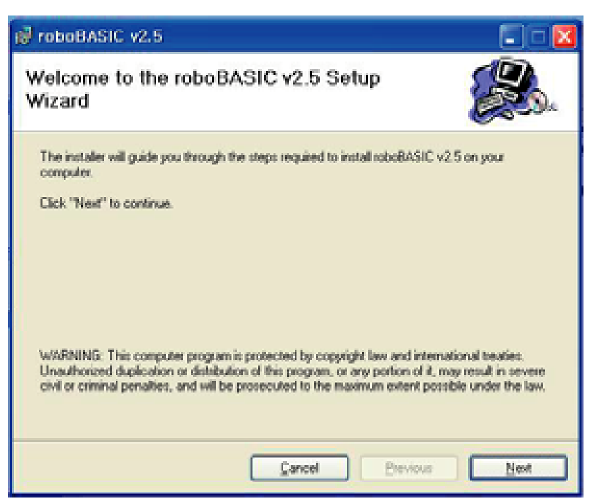
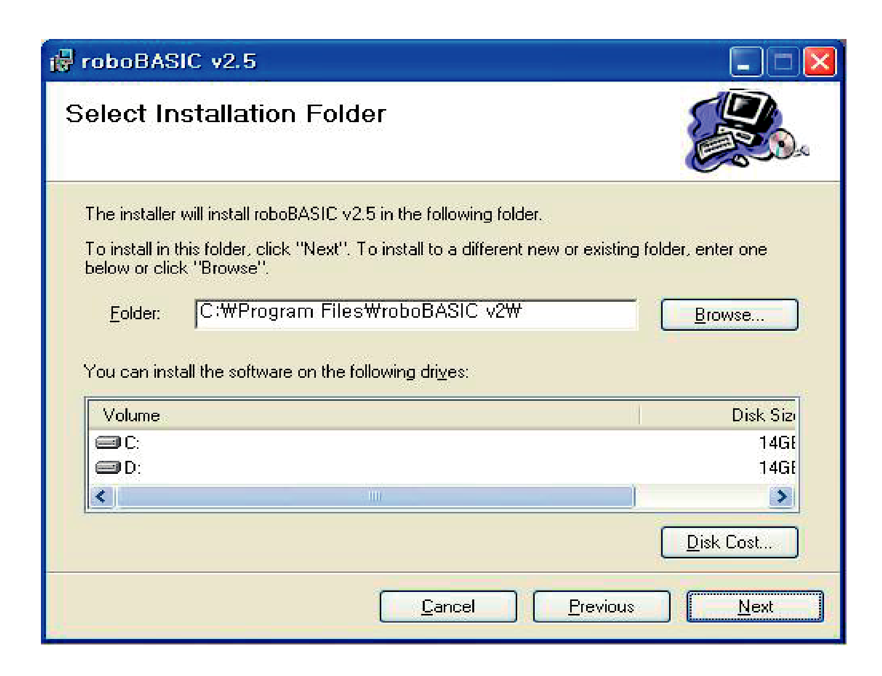


Abbildung 1: Startbildschirm Setup Wizzard

Es erscheint der Startbildschirm des "Setup Wizard", in dem nochmals darauf hingewiesen wird, dass das RoboBASIC nicht ohne ausdrückliche Erlaubnis weitergegeben werden darf. Diesen Bildschirm durch klick auf "Next" bestätigen:

- Auswahl des Installationsverzeichnisses



Im Zweifel ist der vom Programm vorgeschlagene Pfad eine gute Wahl und kann übernommen werden.

- Durch klick auf "Next" kommt man zu einem weiteren Hinweis Bild, in dem das Installationsprogramm die Vollständigkeit aller Angaben bestätigt und den Benutzer dazu auffordert, durch anklicken von "Next" das Einrichten von RoboBASIC einzuleiten.

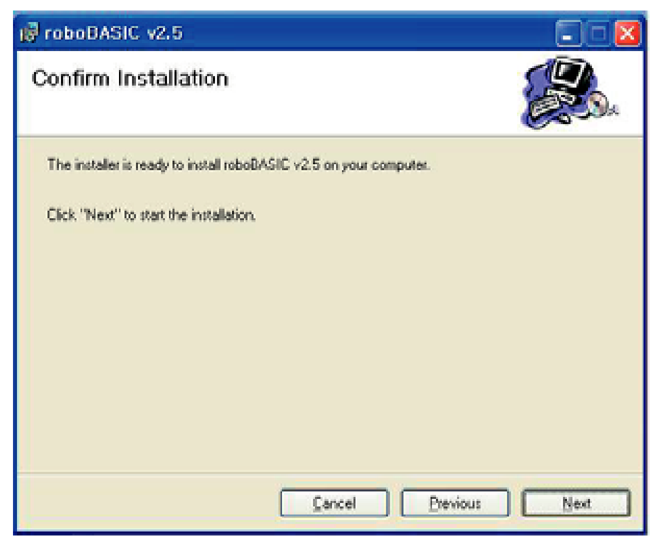
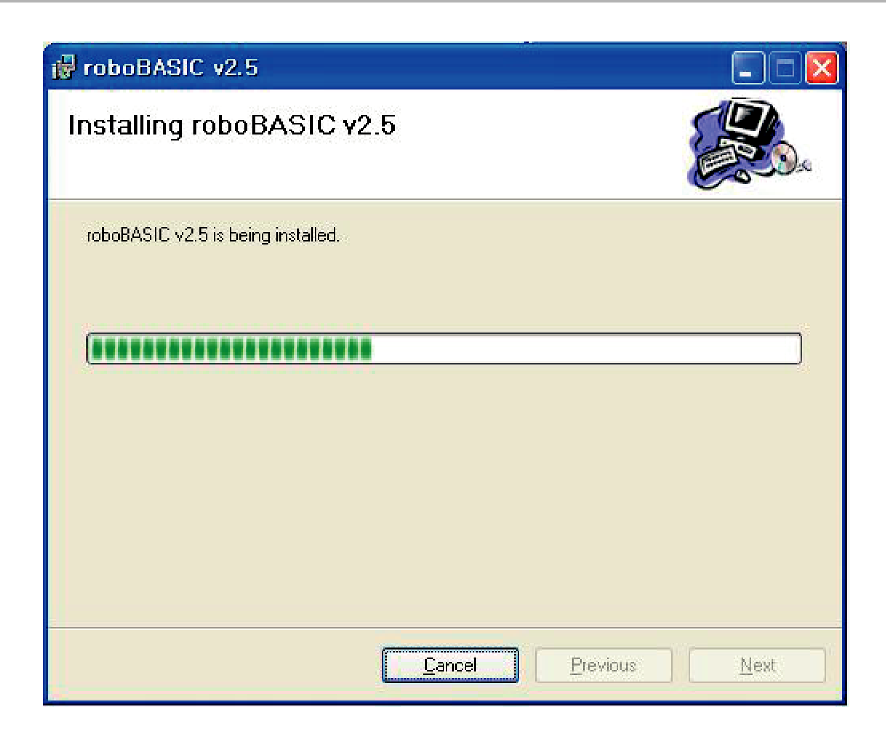


Abbildung 2: Bestätigungsschirm RoboBasic Installation

Der Fortschritt der nun erfolgenden Installation kann an Hand der folgenden Anzeigen verfolgt werden:



Nach vollständiger Installation aller Software erscheint folgende Meldung:

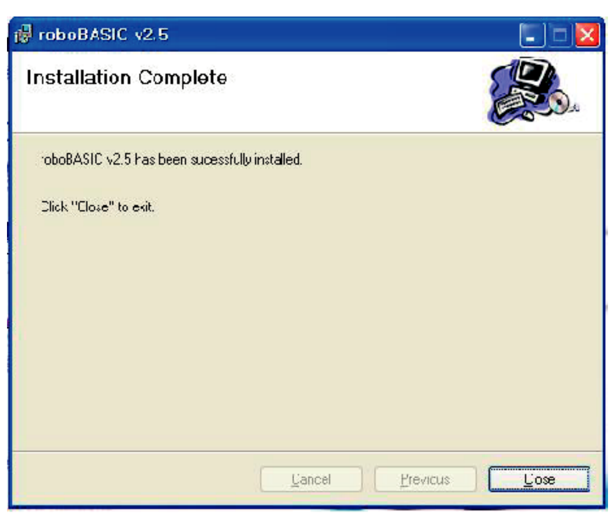


Abbildung 3: Abschlußbildschirm RoboBASIC Installation

Durch klick auf "Close" wird dieses Fenster geschlossen und RoboBASIC, RoboSCRIPT und RoboREMOCON stehen dem Benutzer zur Verfügung, entsprechende Icons wurden automatisch auf dem Desktop angelegt.

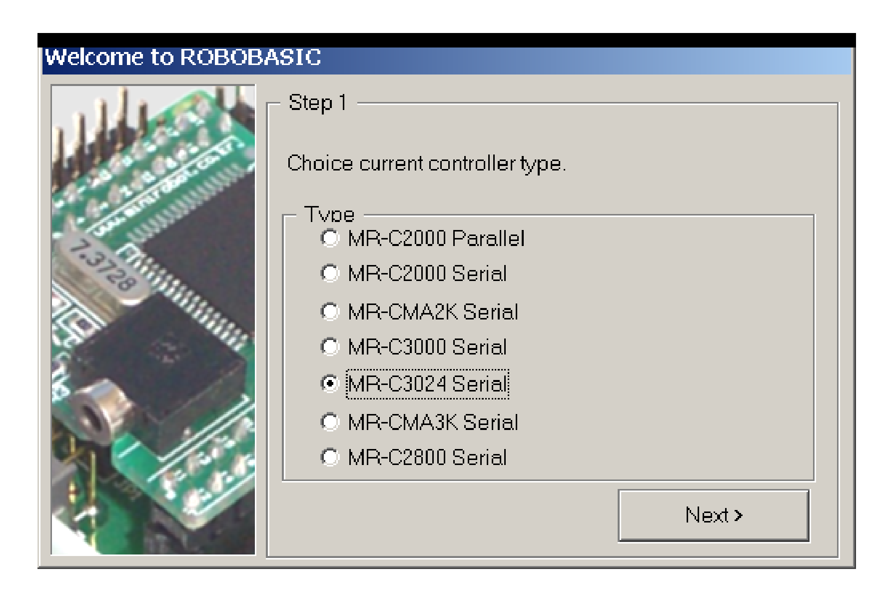
Einrichten von RoboBASIC

Um eine Datenverbindung zum Robot Controller aufbauen zu können, benötigt RoboBASIC die Information, welcher Controller genutzt werden soll und an welchem Port des PCs das Datenkabel

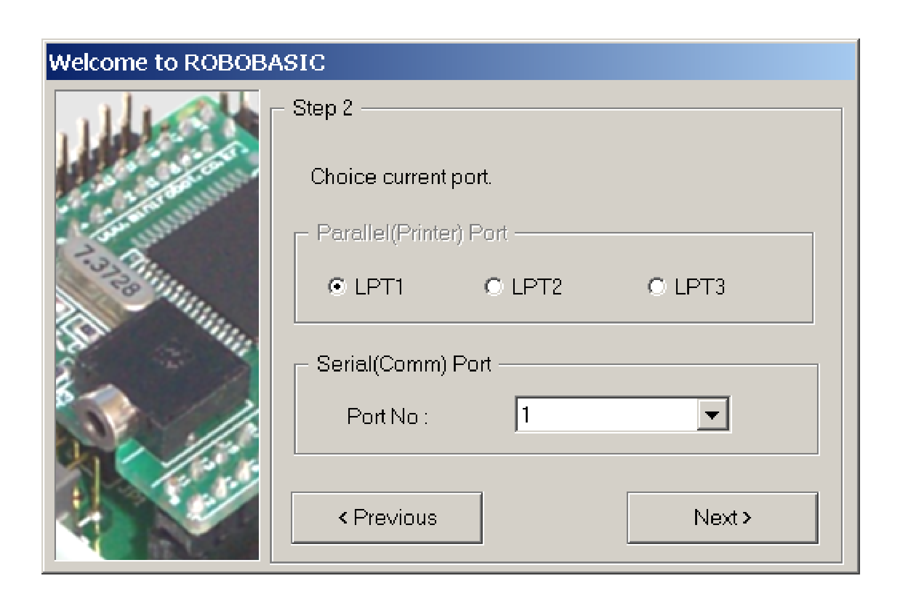
zum Controller angeschlossen wurde.

Verhalten beim Erststart

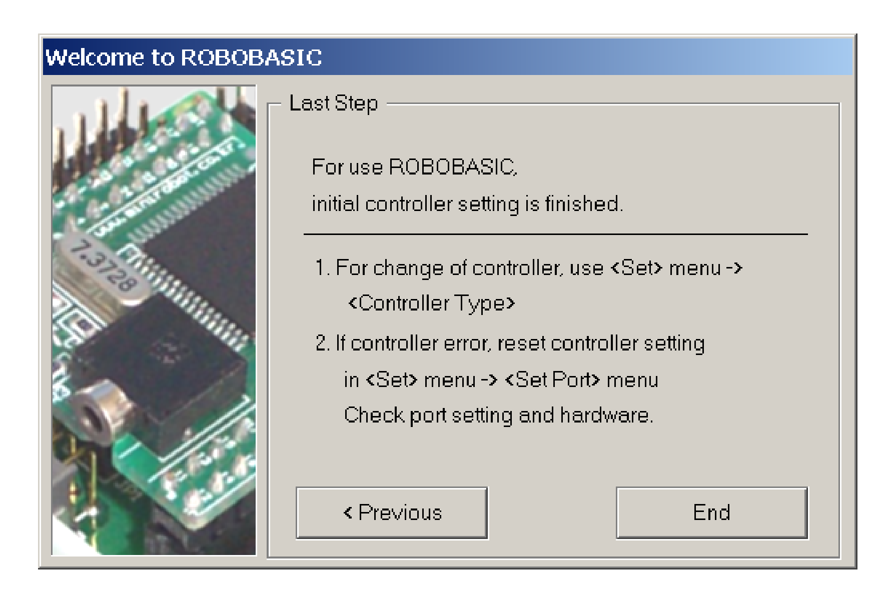
Beim Erststart von RoboBASIC (klick auf das entsprechende Icon) erscheint der Abfragebildschirm, in dem der richtige Controller eingestellt werden kann:



Robonova-1 Set wird standardmässig mit MR-C3024 Controller mitgeliefert.
Daher wählen Sie hier den dem Bausatz beiliegenden Controllertyp MR-C3024



Stellen Sie hier die Schnittstelle, an dem Sie das Datenkabel angeschlossen haben ein. Das dem Bausatz beiliegende Serielle Datenkabel wird im Regelfall an Com 1 (1. serielle Schnittstelle) oder Com 2 (2. serielle Schnittstelle) angeschlossen.



Abschlußbild bei Erstaufruf von RoboBASIC:

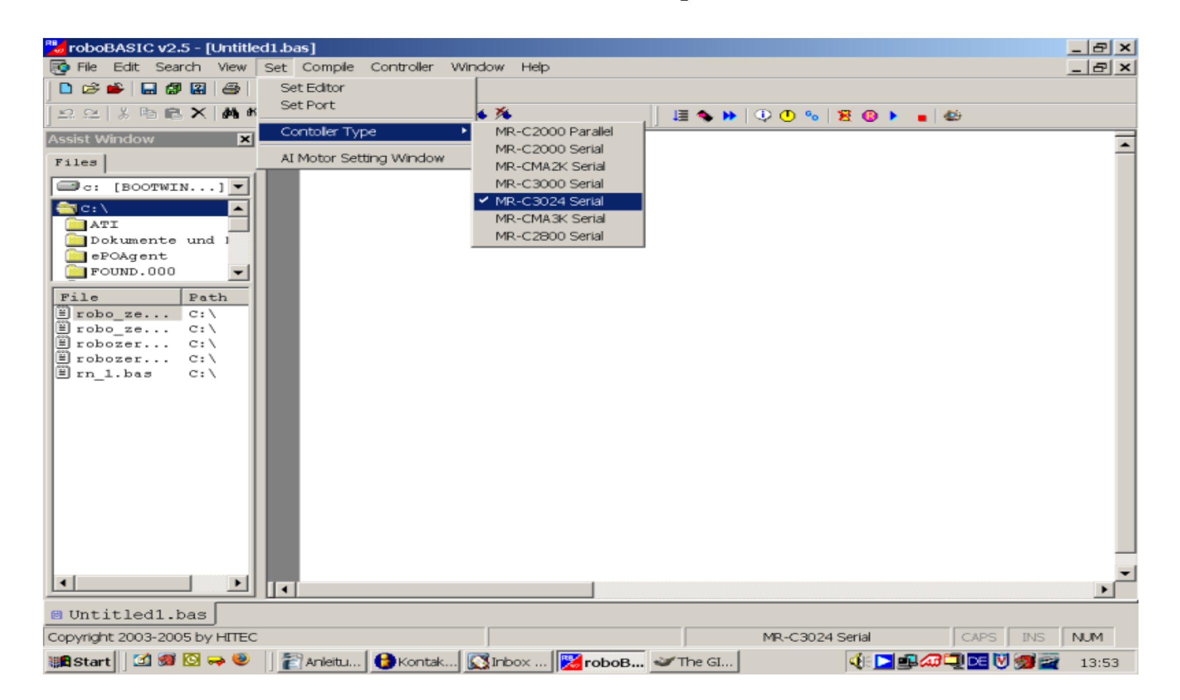
1. Um den Controllertyp anzupassen, wählen Sie im Menü <Set> den Unterpunkt <Controller Type> aus

2 .Sollte es zu Verbindungsfehlern kommen, überprüfen Sie bitte die Einstellungen der Schnittestelle, an dem sich das Datenkabel befindet: Menü <Set> Unterpunkt <Set Port>

Nach bestätigen durch "End" startet die RoboBASIC Programmierumgebung

Anpassung des Controllertyps und der Schnittstelle

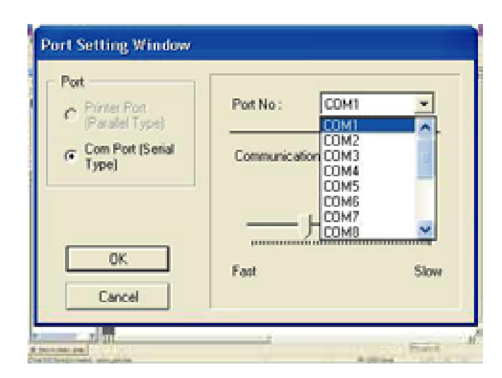
- Starten Sie RoboBASIC durch Klick auf das entsprechende Icon



Wählen Sie im Menü <Set> den Unterpunkt <Controllertype> aus. Markieren Sie mit der Maus den gewünschten Controller (für den im ROBONOVA mitgelieferten Controller ist "MR-C3024" die richtige Wahl).

- Auswahl der PC Schnittstelle für das Datenkabel

- Menü <Set> Unterpunkt <Set Port> auswählen :



- Wählen Sie die Serielle Schnittstelle (Com Port) aus, an dem Sie das Datenkabel zum Controller angeschlossen haben.

Überprüfen der Verbindung PC - Roboter

Hinweis: Ohne eine funktionierende Datenverbindung ist keine Programmierung oder Einstellung des Roboters möglich!

- Schließen Sie das Datenkabel an die von Ihnen gewählte PC Schnittestelle an
- Verbinden Sie den Klinkenstecker des Datenkabels mit der entsprechenden Buchse des Robot Controllers



Abbildung 4: Anschluß Datenkabel an Robotercontrollboard

- Starten Sie RoboBASIC
- Schalten Sie das Controllerboard ein
- Klicken Sie auf das Icon "!" in der Icon Zeile des RoboBASIC Editor

(Alternativ können Sie auch im Menü <Controller> Unterpunkt <Controller Information> anwählen)

Bei erfolgreichem Datenaustausch zwischen Controller und PC werden Ihnen in einem Fenster die Informationen zum Controller angezeigt:

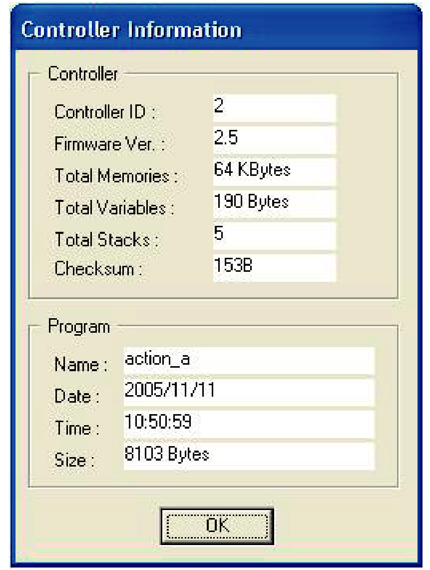


Abbildung 5: Bei erfolgreichem Datenaustausch zwischen Controller und PC werden Ihnen in einem Fenster die Informationen zum Controller angezeigt

Hinweis: Die angezeigten Inhalte können je nach Controller und geladenem Programm von den hier gezeigten verschieden sein, wichtig ist, dass überhaupt Informationen angezeigt werden.

Fehlermeldungen

Sollte stattdessen eine Fehlermeldungen (meist "Error in data transmission") erscheinen und das Controller Informationsfenster leer bleiben, konnte keine Datenverbindung zum Controller aufgebaut werden.

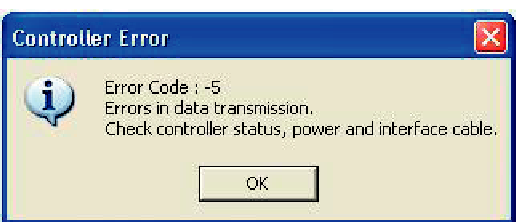


Abbildung 6: Fehler in Datenübertragung

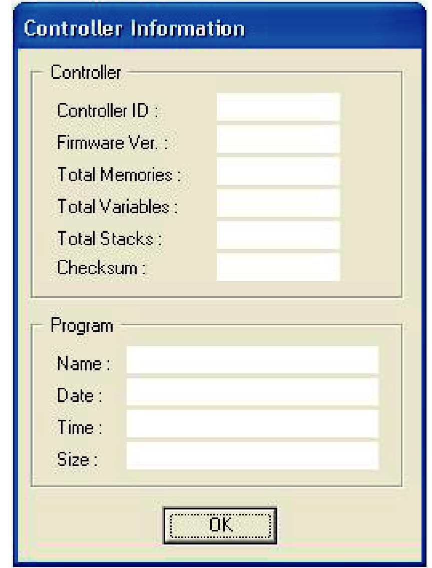


Abbildung 7: Leeres Controller Informations Fenster deutet auf fehlerhafte Datenverbindung hin !

Checkliste bei fehlerhafter Datenverbindung:

- Akku ausreichend geladen ? (ggf. Akku vollständig laden)
- Ist als Controller der Richtige Typ angewählt? (Für ROBONOVA MR-C3024!)
- Richtiger COM Port gewählt? (alternativ andere Ports testen)
- Verbindungskabel vollständig eingesteckt?
- Sind andere Programme aktiv, die ggf. auch auf einen Com Port zugreifen können und diesen blockieren? Im Zweifel alle anderen Anwendung schließen und Windows neu booten.

Programmierung in RoboBASIC

Nach erfolgreichem Überprüfung der Datenkommunikation zwischen PC und Controller können Sie nun Ihre ersten Schritte zur Programmierung des Controllers durchführen:

Dazu muss der Quelltext des Programms im Editor vorliegen:

- Sie können ein eigenes neues Programm erstellen
  - Quelltext in Editorfenster eingeben und abspeichern
  - Bei Bedarf neues Editor Fenster über Menü <File> Unterpunkt <New Program File> öffnen.
- Sie können eines der beigefügten Beispielprogramme in den Editor laden:
  - Menü <File> Unterpunkt <Open Program File> anwählen
  - Suchen Sie im Dateiauswahlmenü nach Beispieldateien, RoboBASIC Dateien werden mit der Erweiterung ".bas" abgespeichert. Wählen sie ein Beispielprogramm aus und laden es in den Editor

Um aus dem Quelltext ein lauffähiges Programm zu erzeugen, muss dieser in eine durch den

Controller verstandenes Format (den sog. Object Code) übersetzt werden. Diesen Vorgang bezeichnet man auch als compilieren. Der Object Code wird anschließend über die Datenverbindung zum Controller übertragen (sog. Download) und kann dann dort ausgeführt werden.

Compilieren des Quelltextes

- Auswahl des Icons aus der Menüsymbolleiste
- alternativ: Menü <Compile> Unterpunkt <Make Object Code>
- alternativ: F2 Taste (Shortcut)

Der Compiler führt zuerst eine Syntaxüberprüfung des Quelltextes durch, werden Fehler erkannt, bricht der Compiler mit einer entsprechenden Fehlermeldung ab.

Wird kein Syntaxfehler erkannt, erscheint ein Fortschrittsbalken,

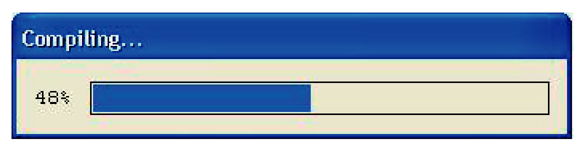


Abbildung 8: Programmübersetzung zu 48 % abgeschlossen

der bei erfolgreicher Programmübersetztung, nach erreichen der 100% Marke, wieder ausgeblendet wird.

Erscheint keine Fehlermeldung, war die Übersetzung der Quelltextes erfolgreich und der erzeugte Object Code kann zum Controller übertragen werden:

Download des Object Codes zum Controller

- Auswahl des Icons aus der Menüsymbolleiste
- alternativ: Menü <Compile> Unterpunkt <Download> auswählen
- alternativ: F6 Taste (Shortcut)
  Das Download Fenster gibt zuerst die Größe des Object Codes an, die Datenübertragung zum Controller startet nach Bestätigung durch den "ok" Knopf.

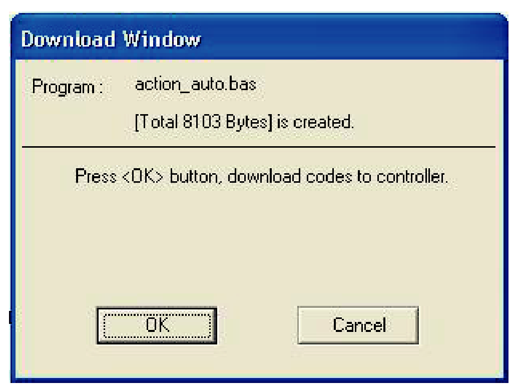


Abbildung 9: OK Knopf drücken, um Programm zum Controller zu übertragen

Der Fortschritt der Datenübertragung wird im Download Fenster angezeigt:

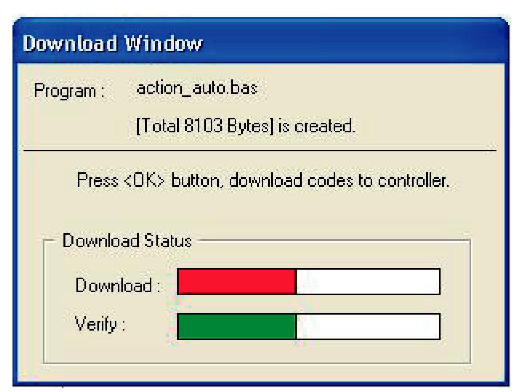


Abbildung 10: Fortschrittsanzeige bei der Datenübertragung zum Controller

Compilieren und Download in einem Arbeitschritt

- Auswahl des Icons aus der Menüsymbolleiste
- alternativ: Menü <Compile> Unterpunkt <Run All> auswählen
- alternativ: F9 Taste (Shortcut)

RoboBASIC - Setzen der Servo Nullpunkte

Auch wenn der Zusammenbau des Roboters fehlerfrei vorgenommen wurde, kann es nach der Erstinbetriebnahme zu geringen Abweichungen der Neutrallage der Servos kommen. Dieses ist nicht ungewöhnlich und kann mittels des nachfolgend beschriebenen Abgleichverfahrens ausgeglichen werden. Ein sorgfältiger Neutralabgleich ist die Voraussetzung zur Nutzung der beigefügten Musterbewegungen.

Der Abgleich der Neutralposition erfolgt dabei unter Zuhilfenahme der RoboBASIC Programmierumgebung.

- Starten die RoboBASIC und Verbinden den Controller mit dem PC
- Wählen Sie im Menü < Compile> den Unterpunkt < Set Zero Point> aus



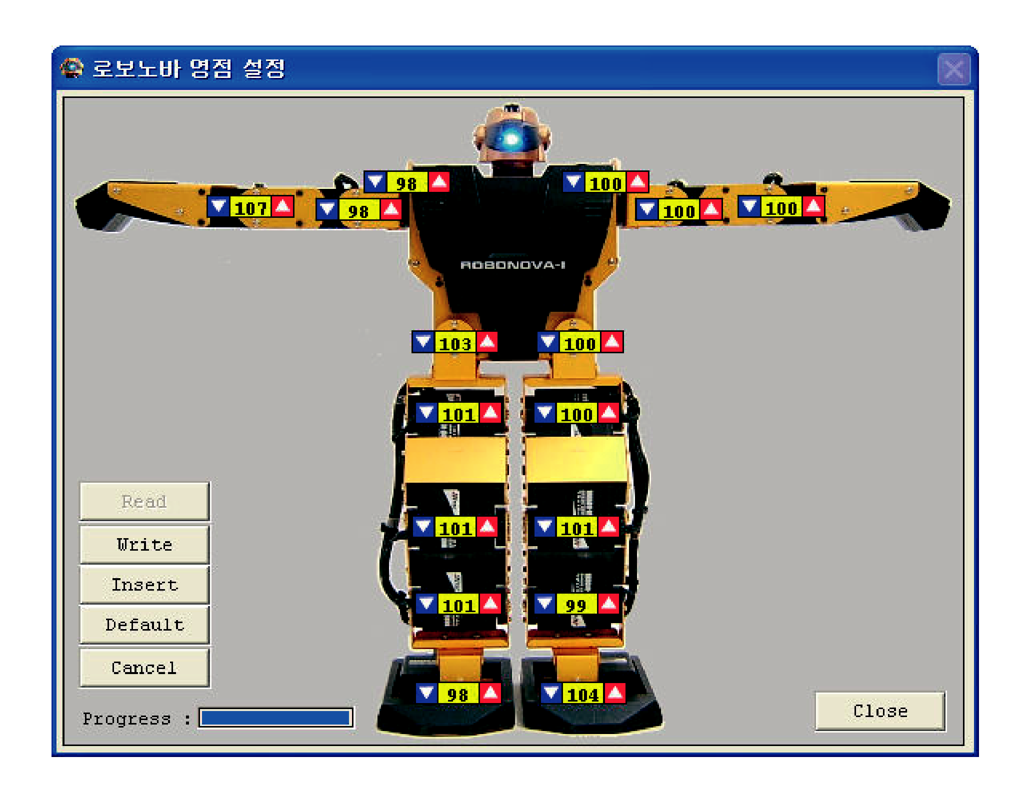
Es erscheint ein Fenster mit ROBONOVA in Frontalansicht , wobei für jedes Servo eine Einstellmöglichkeit für die Neutralstellung vorhanden ist.

Ziel des Abgleichs ist es, alle Servos so einzustellen, dass der Roboter wie abgebildet gerade ausrichtet auf einer ebenen Unterlage steht. (Details bitte unbedingt aus nachfolgenden Abbildungen entnehmen!)

- **<u>ACHTUNG:</u>** Stellen Sie den Roboter zuerst auf eine ebene Unterlage und betätigen sie erst

dann den READ Knopf. Vermeiden Sie es, den Roboter in der Hand zu halten, da unmittelbar nach dem READ Befehl alle Servos in die Neutralposition laufen und so u.U. Verletzungen an der Hand auslösen können!

- Clicken sie auf den READ Knopf, um den Roboter in seine aktuell eingestellte Neutrallage zu bringen:



- Nehmen Sie nun die Feineinstellung aller Servos vor, in dem sie die "hoch" / "runter" Pfeile für das dem jeweiligen Gelenk zugeordneten Servo nutzen, um dieses in die Sollposition zu fahren.

Weitere Detailansichten zur Nullpunkltausrichtung:

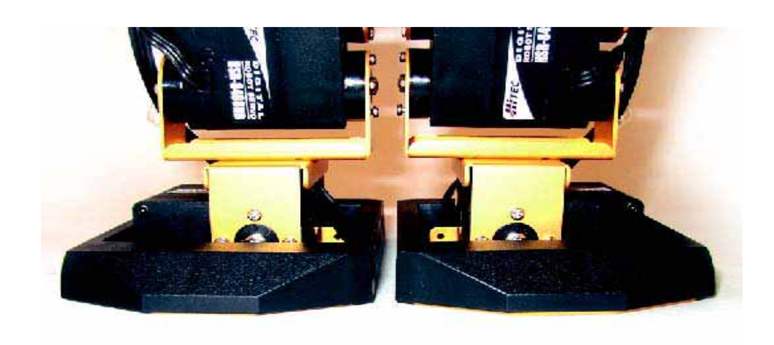
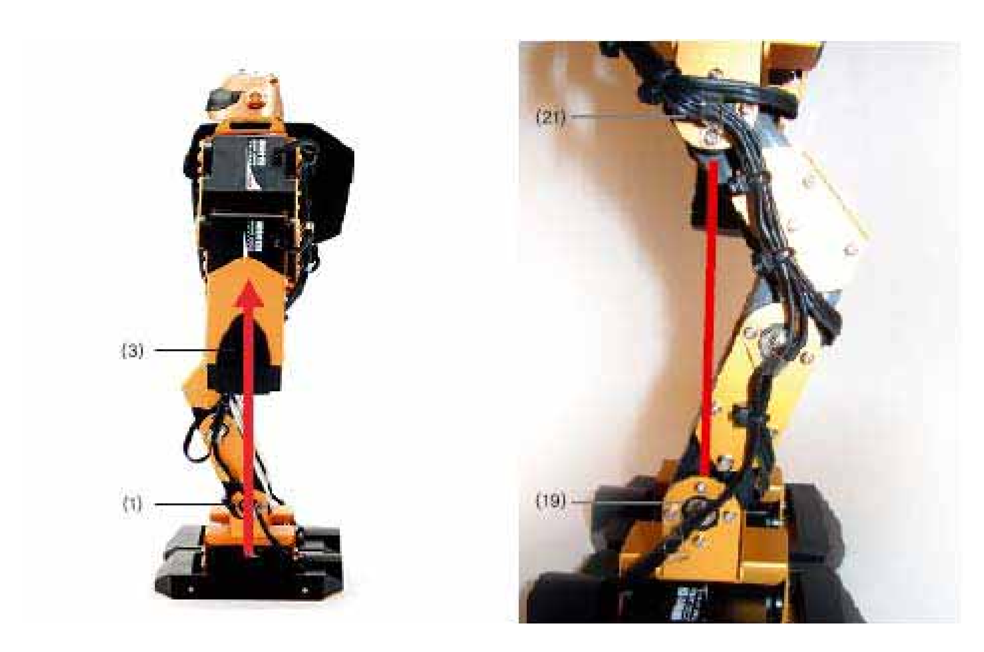


Abbildung 11: Achten Sie darauf, dass beide Füße parallel zueinander auf der Unterlage aufliegen!



Servo Nr 1. und Nr.3 des linkes Beins und Servo Nr.19 und Servo Nr.21 des rechten Beines müssen eine gerade Linie bilden.



Abbildung 12: Ausrichtung der Beine

Der Abstand zwischen linken und rechtem Bein muss ca. 8 mm betragen

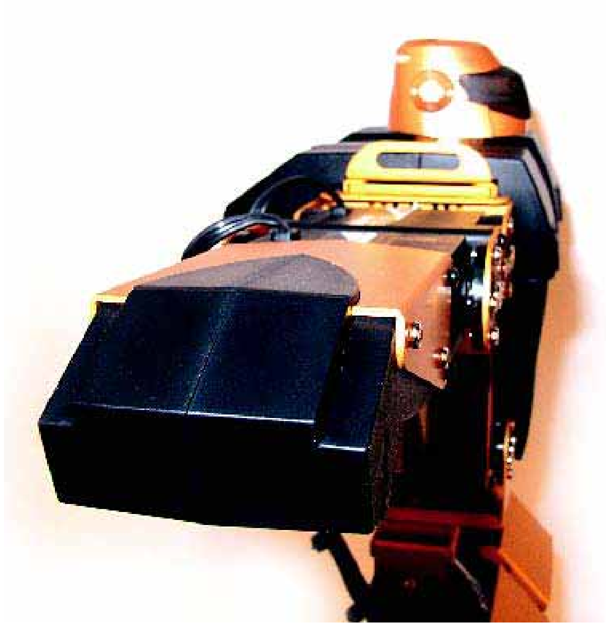


Abbildung 13: Ausrichtung der Arme

Die Arme, wie hier gezeigt, in einer geraden, horizontalen Linie ausrichten

Abspeichern der Neutrallage

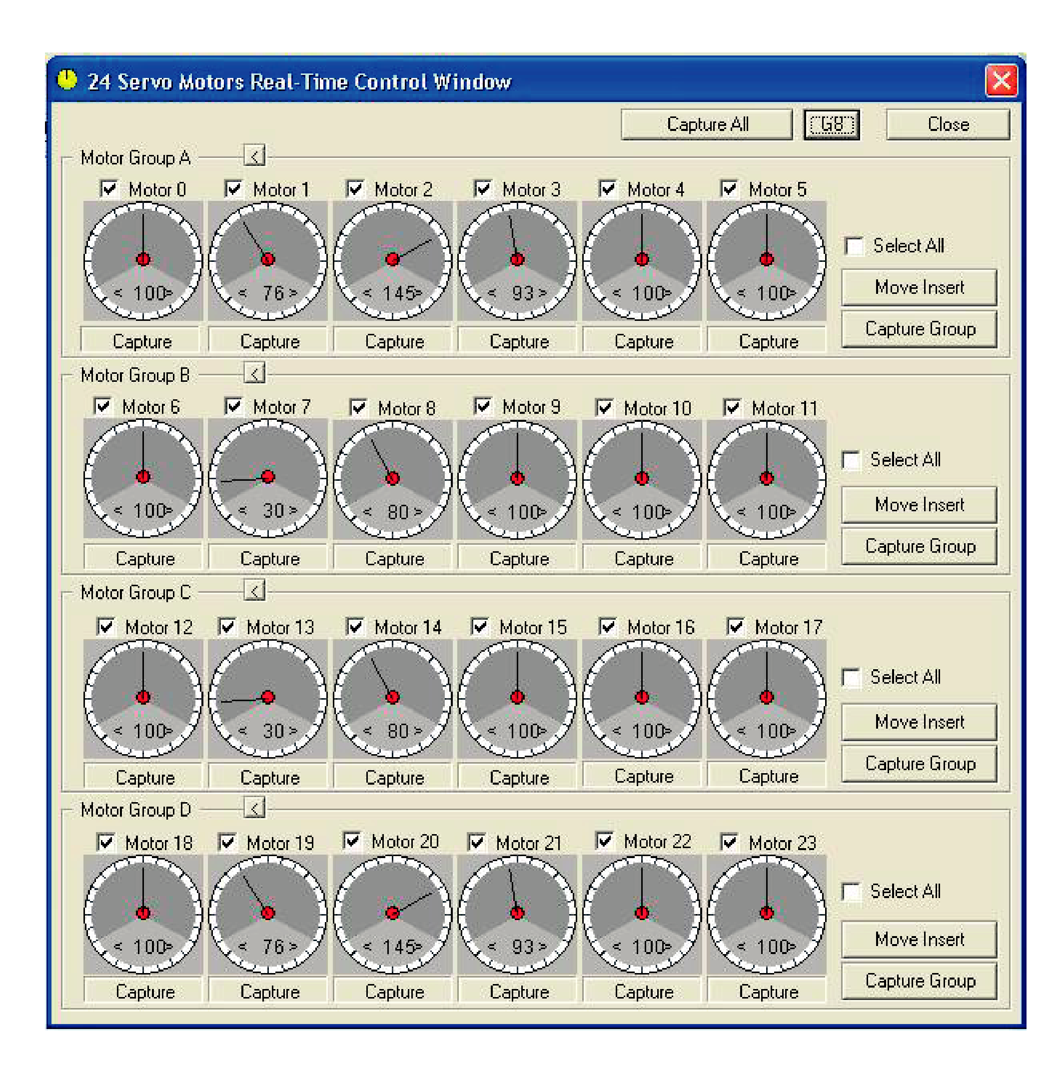
- Um die gefunden Neutrallage in Ihren Quelltext einzufügen, wählen Sie den "INSERT" Knopf
- Um die gefundene Neutrallage direkt in dem Speicher von MR-C3024 abzuspeichern, wählen die "WRITE" Funktion aus. Hier werden die gefundenen Werte direkt zum Roboter übertragen und dort dauerhaft abgespeichert. Um diese neuen Werte zu überprüfen, schalten Sie den Roboter aus und wieder ein.

Servo Positionen in Echtzeit ändern

Mit Hilfe des in RoboBASIC enthaltenden "Real Time Control Windows" können Sie direkt jedes Servo einzeln ansprechen, in die gewünschte Position bewegen oder eine manuell vorgegebe Position auslesen und in Ihr RoboBASIC Programm übernehmen.

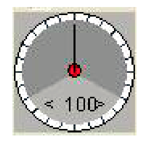
- Starten Sie die RoboBASIC Programmierumgebung und verbinden Sie PC und Controller mit dem Datenkabel
- Start der Echtzeit Servokontrolle durch Klick auf das Symbol in der Symbolleiste
  - alternativ: Aufruf über das Menü <Controller> Unterpunkt <Servo motor realtime control>
  - alternativ: Aufruf über F7 Taste (Shortcut)

  Hinweis: Nach Start dieses Programmteils wird eine Datenverbindung zum Controller aufgebaut und die Servoinformationen zum PC übertragen. Bei Fehlermeldungen unbedingt die Verbindung PC- Roboter überprüfen.



Nach erfolgreichem Auslesen der Servopositionen öffnet sich das Real-Time Control Fenster

Motor 0 Servo Motor 0 ist aktiv, das Servo hält aktiv die eingestellte Position
Durch Klick in das Markierfeld wird der Servomotor stromlos geschaltet, das betreffende Servo kann jetzt manuell in die neue Position bewegt werden. Ein erneuter Klick auf das Markierfeld aktiviert den Servomotor wieder, die neue Position wird dann vom Servo gehalten .



Positionsanzeige des jeweiligen Servos, Wertebereich von 10 bis 190

G6 Schaltet Anzahl der Servos pro Servogruppe um
: G6 = Gruppen zu 6 Servos in einer Zeile (für ROBONOVA empfohlene Darstellung)
: G8 = Gruppen zu 8 Servos in einer Zeile

Capture Group Capture Funktion: Hier werden die Servopositionen aktualisiert und neu eingelesen

Move Insert Fügt einen Move Befehl in den im Editorfenster geöffneten Quelltext an der Stelle des Cursors ein und erlaubt so die Übernahme der aktuellen Servopositionen direkt in Ihr Basicprogramm.
Achtung: Es werden nur die aktiven Servos übernommen, nicht markierte Servopositionen werden unverändert gelassen.

Beispiel für erzeugten Code : MOVE G8A,100,76,145,93,100,100,100,100 (alle Servos aktiv)
MOVE G8A,100, , ,93,100,100,100,100 (Servo 1,2 nicht aktiv)

Servopositionen direkt in Quelltext einfügen

RoboBASIC stellt zwei Werkzeuge zur einfachen Erstellung von Bewegungsabläufen zur Verfügung:

1. Echtzeit Servocontrol Fenster
2. ROBONOVA Servocontrol Fenster

Nutzung des Echzeit Servocontrol Fensters

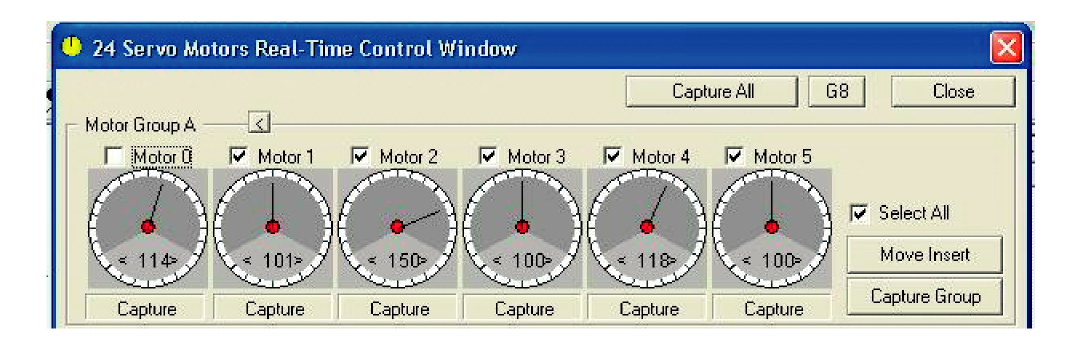
- Starten Sie die RoboBASIC Programmierumgebung und verbinden Sie PC und Controller mit dem Datenkabel
- Laden Sie Ihren Quelltext in den Editor oder erstellen Sie ein neues Grundprogramm
- Positioniern Sie den Cursor des Editors auf die Stelle im Quelltext, wo die Servobewegungen(MOVE -Befehl) eingefügt werden sollen
- Start der Echtzeit Servokontrolle durch Klick auf das Symbol in der Symbolleiste
  - alternativ: Aufruf über das Menü <Controller> Unterpunkt <Servo motor realtime control>
  - alternativ: Aufruf über F7 Taste (Shortcut)
- <u>**Hinweis:**</u> Nach Start dieses Programmteils wird eine Datenverbindung zum Controller aufbaut und die Servoinformationen zum PC übertragen. Bei Fehlermeldungen unbedingt die Verbindung PC- Roboter überprüfen.

Nun können die gewünschten Servopositionen auf zwei Arten eingestellt werden:

Manuelle Einstellung der Servopositionen:

Hier können sie alle zuvor auf Freilauf geschalteten Servos von Hand in die Wunschposition bewegen, dieses Verfahren ist bei Robotern sehr zu effektiv in der Anwendung, da bei den meisten Bewegungen mehr als ein Servo beteiligt ist.

Vorgehensweise



- Schalten Sie die, für die Bewegung, benötigten Servos durch klick auf das Markierungsfeld

auf Freilauf (der Motor des Servos wird hier stromlos geschaltet) Im obigen Beispiel ist der Servo Motor 0 im Freilaufmodus.

- Bewegen Sie die im Freilaufmodus befindlichen Servos in die gewünschte Positionen
- Klicken Sie den Capture Knopf an, um die neue Servoposition einzulesen
- Klicken Sie erneut in das Markierfeld des im Freilauf befindlichen Servos, der Servomotor wird wieder mit Strom versorgt und hält nun aktiv die neue Position.
- Um diese Servostellungen in Ihren Quelltext einzuzufügen, betätigen sie den <MOVE INSERT> Knopf. Hinweis: In der erzeugten MOVE Befehlszeile werden nur die Servos aus der jeweiligen Gruppe berücksichtigt, die als aktiv (Servo hält Position) gekennzeichnet sind. Alle Servos im Freilauf werden mit Lehrzeichen anstelle des Position zurückgeliefert.

Einstellung der Servoposition mittels Maus oder Pfeiltasten

Warnung! Bei direkter Einstellung der Servos können die Servos unter Umständen durch versehentliches Verstellen sehr heftige Bewegungen ausführen

- Aktivieren Sie den gewünschten Servomotor ☑ Motor 0 durch setzen des Markierfeldes
- Klicken Sie mit der Maus in die Positionsanzeige
- Bei gedrückter rechter Maustaste können Sie den Zeiger auf die gewünschte neue Position stellen, nach loslassen der Maustaste nimmt das Servo sofort die neue Position ein.
- Nutzen sie die Pfeil links / Pfeil rechts Tasten um die Servoposition in kleinen Schritten zu ändern.
- Um diese Servostellungen in Ihren Quelltext einzuzufügen, betätigen sie den <MOVE INSERT> Knopf. Hinweis: In der erzeugten MOVE Befehlszeile werden nur die Servos aus der jeweiligen Gruppe berücksichtigt, die als aktiv (Servo hält Position) gekennzeichnet sind. Alle Servos im Freilauf werden mit Leerzeichen anstelle des Position zurückgeliefert.

Tipp: Nutzen Sie das <Select all> Markierfeld, wenn sie alle Servos einer Gruppe gleichzeitig ändern möchten.

ROBONOVA Servo Control Fenster

Das ROBONOVA Servo Control Programm ist Teil der RoboBASIC Programmierumgebung und ist eine, speziell am Robnonova ausgerichtete, Variante des allgemeineren Servo Control Programmes. Die Servogruppen können hier besonders anschaulich ausgewählt werden. Weiterhin stehen Funktionen zum erzeugen symmetrischer Servobewegungen zur Verfügung.

Aufruf ROBONOVA Servo Control:

- Wählen Sie im RoboBASIC Menü <Controller> Unterpunkt <ROBONOVA motor control> aus
- Es werden die aktuellen Servopositionen vom Roboter zum PC übertragen, bei Fehlermeldungen bitte die Verbindung PC - Roboter überprüfen

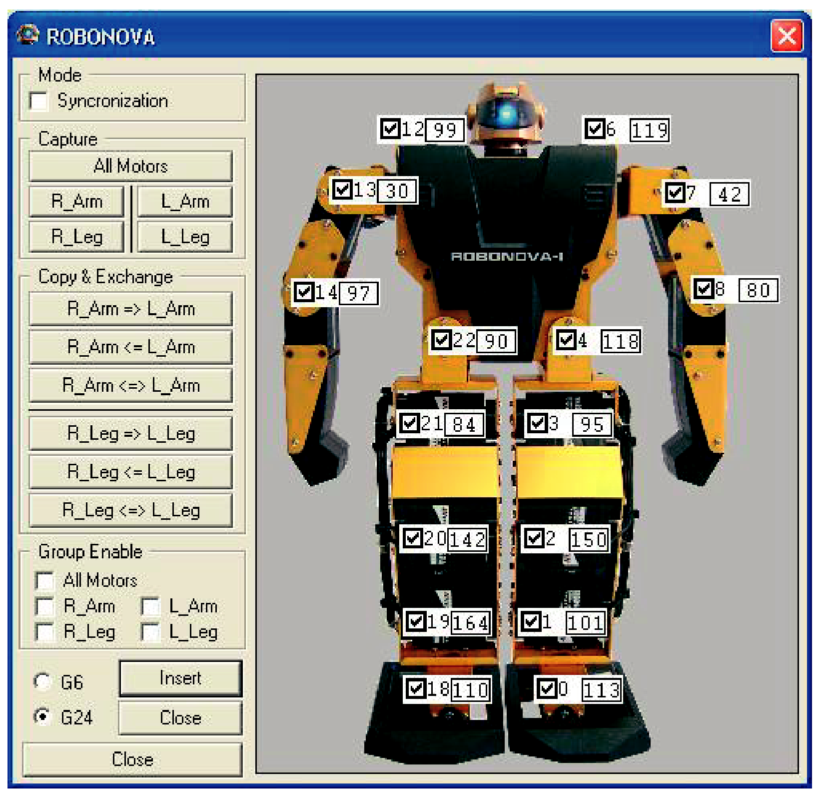


Abbildung 14: ROBONOVA Servo Control Fenster

Ausführen einer Einzelnen Servobewegung direkt aus dem Quelltext

RoboBASIC unterstützt das direkte zeilenweise testen von Einstellungen, wie sie durch die Befehle OUT, MOVE, SERVO, POSE und MOVEpose ausgelöst werden, ohne das gesamte Programm compilieren und downloaden zu müssen.

Vorgehensweise:

- Laden oder erstellen Sie ein RoboBASIC Programm im Editor.
- Stellen Sie eine Datenverbindung zum Controller her
- Platzieren Sie den Cursor auf die Zeile, die Sie testen möchten.
- Rufen Sie im Menü <Controller> den Unterpunkt <Direkt Line Control> auf.
    - alternativ: F5 Taste (Shortcut)

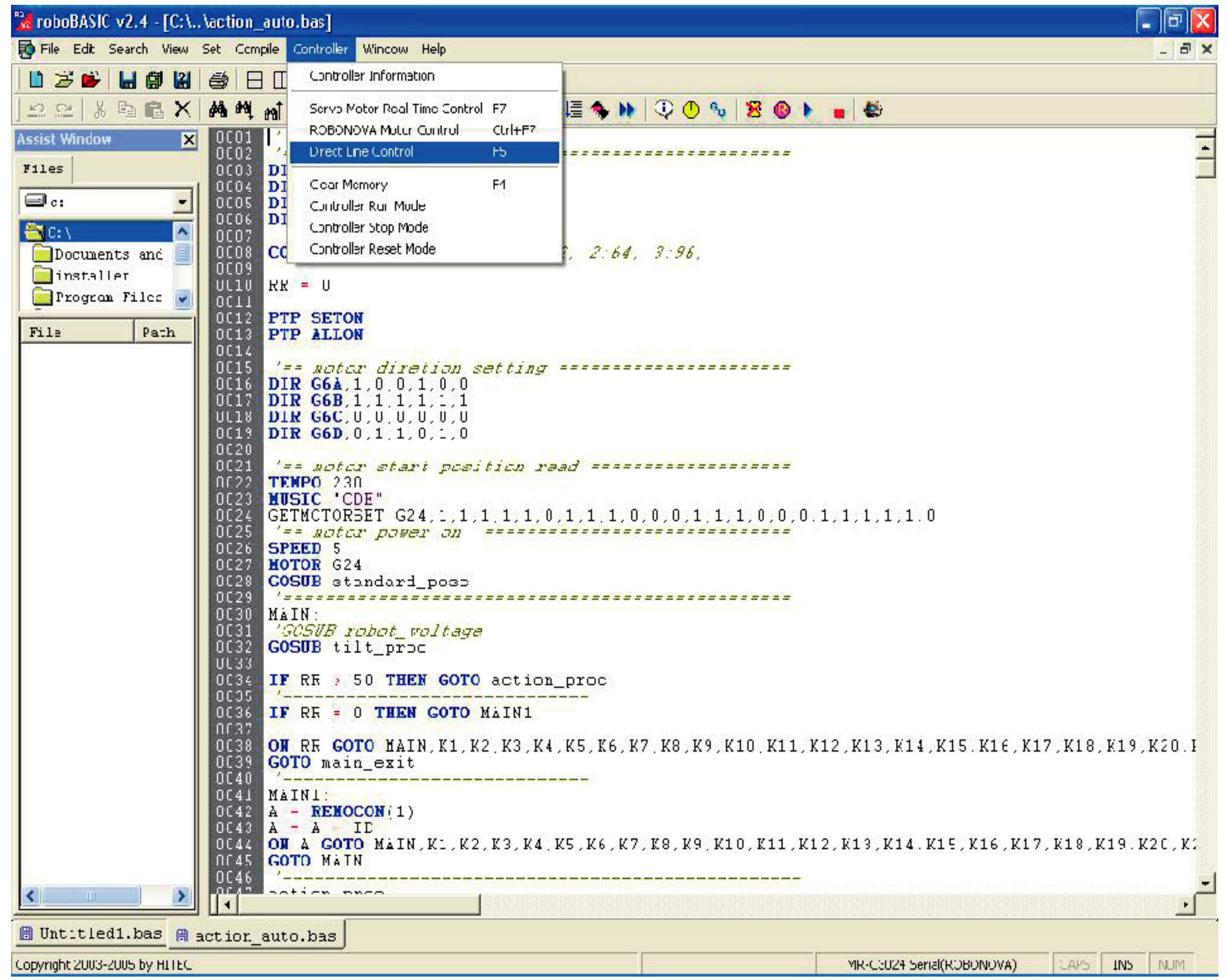


Abbildung 15: Ausführen einer Einzelnen Servobewegung direkt aus dem Quelltext

Beachten Sie, dass die betreffende Zeile unmittelbar ausgeführt wird!

Nutzung von roboScript 2.5

Information

Diese Anleitung bezieht sich auf roboScript v 2.5. An Software und Handbuch können zur Verbesserung ohne Bekanntgabe Änderungen vorgenommen werden.

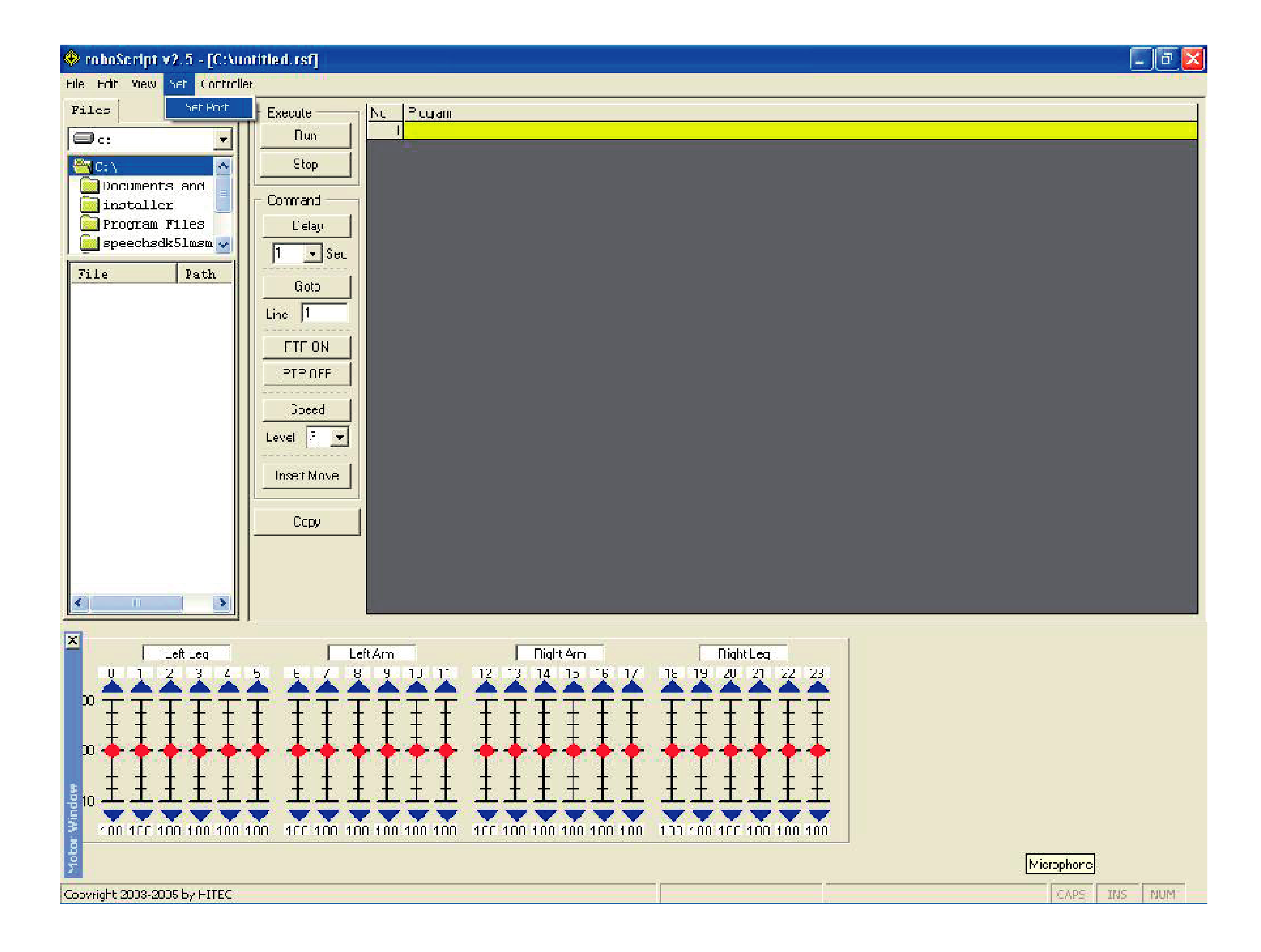
RoboScript ist eine registrierte Software. Erstellen von Kopien, Veröffentlichung, Versand oder Vertrieb dieser Anleitung oder der Software ohne Genehmigung ist untersagt.

Installation

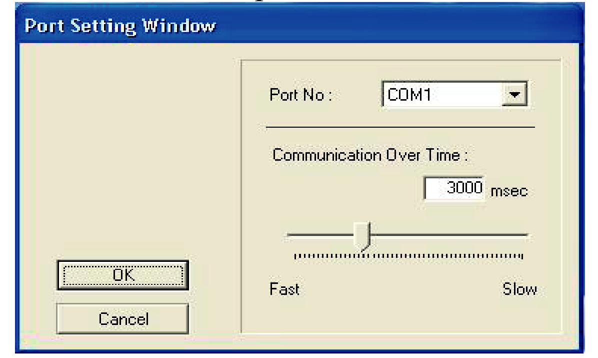
RoboScript wird bei der Installation von RoboBASIC ab der Version 2.5 automatisch mit installiert.

Einrichtung von roboScript

- Schließen Sie das Datenkabel an eine serielle Schnittstelle Ihres PC an
- Verbinden Sie das Datenkabel mit dem Controller
- Schalten Sie den Controller ein.
- Starten Sie roboScript durch Klick auf das entsprechende Icon auf Ihrem Desktop



- Passen Sie die PC Schnittstelle für das Datenkabel an:
  - Menü <Set> Unterpunkt <Set Port> auswählen

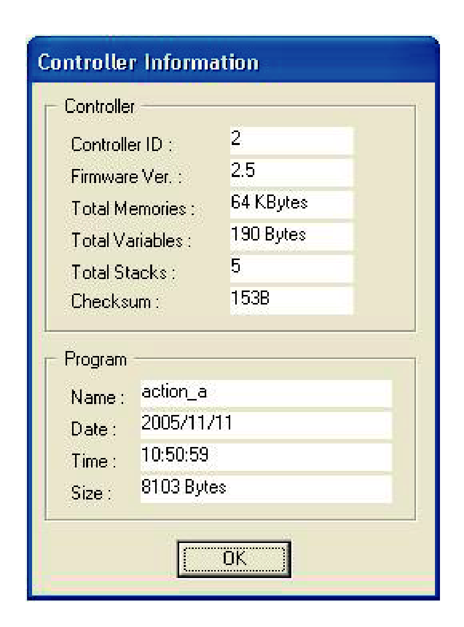


  - Wählen Sie im Feld Port No: die Schnittstelle, mit der das Datenkabel verbunden ist, aus und bestätigen mit <ok>.

Überprüfen der Datenverbindung

- Wählen Sie im Menü <Controller> den Unterpunkt <Contoller Information> aus.

Bei erfolgreichem Datenaustausch zwischen Controller und PC, werden Ihnen in einem Fenster, die Informationen zum Controller angezeigt:



Hinweis: Die angezeigten Inhalte können je nach Controller und geladenem Programm von den hier gezeigten verschieden sein, wichtig ist, dass überhaupt Informationen angezeigt werden.

Fehlermeldungen



Sollte stattdessen eine Fehlermeldungen erscheinen und das Controller Informatinonsfenster leer bleiben, konnte keine Datenverbindung zum Controller aufgebaut werden.

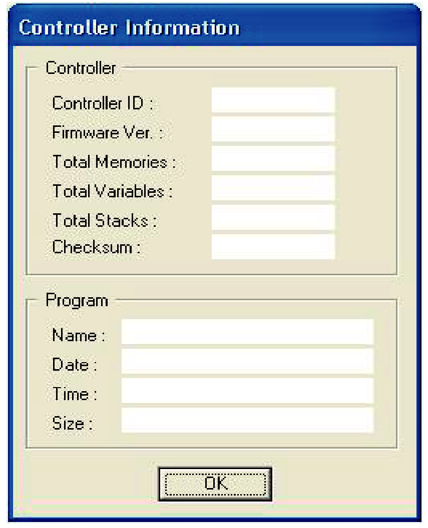


Abbildung 16: Leeres Controller Informations Fenster deutet auf fehlerhafte Datenverbindung hin !

Checkliste bei fehlerhafter Datenverbindung:

- Akku ausreichend geladen? (ggf. Akku vollständig laden)
- Richtiger COM Port gewählt? (alternativ andere Ports testen)
- Verbindungskabel vollständig eingesteckt?
- Sind andere Programme aktiv, die ggf. auch auf einen Com Port zugreifen können und diesen blockieren? Im Zweifel alle anderen Anwendung schließen und Windows neu booten.

Programmierung unter roboScript

RoboScript dient zum einfachen Erstellen von Bewegungsabläufen. Die Nutzung von RoboScript setzt eine funktionierende Datenverbindung PC- Robot Controller voraus, da das Programm mittels Datenkabel einmalig zum Robot Controller übertragen wird.

Erste Programmierschritte

- Starten Sie RoboScript und stellen die Datenverbindung zum Roboter her

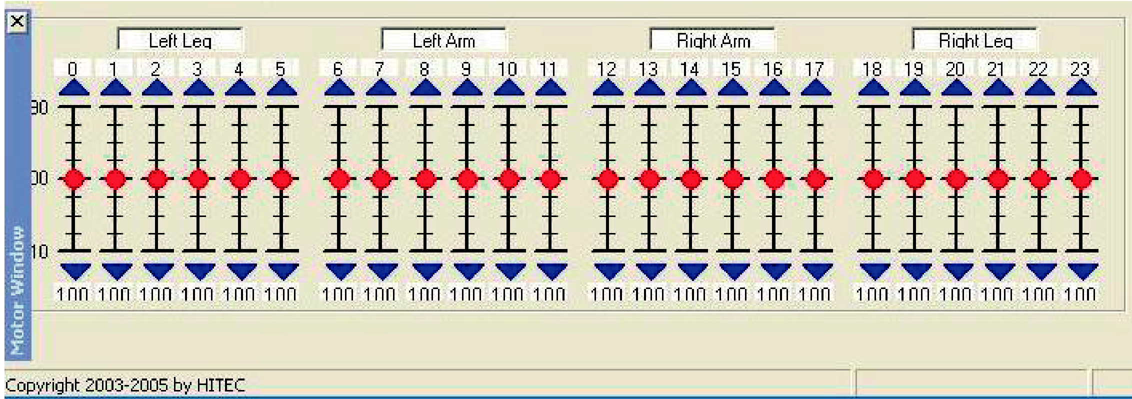


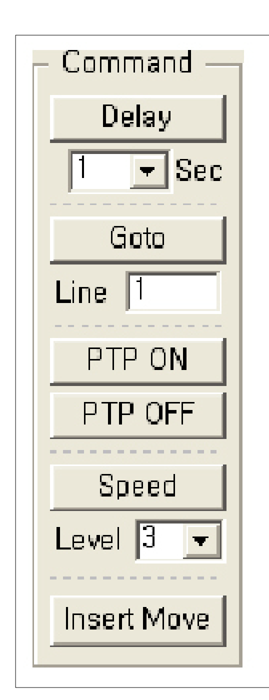
Abbildung 17: Die Servopositionen werden im Motor Window mit Hilfe der Schieberegler bestimmt

- Bewegen Sie hierzu die Maus über den roten Punkt des jeweiligen Schiebereglers und

drücken die rechte Maustaste, um die Position zu verändern. Die Änderung wird erst beim Loslassen der Maustaste an das Servo übertragen.

- Bearbeiten Sie so jede einzelne Servoposition, bis sie den Roboter in die richtige Position gebracht haben.

- Durch einen Mausklick auf die <Insert Move> Taste wird aus den aktuell über die Schieberegler eingestellten Servopositionen eine neue Zeile RoboScript erzeugt.

	Delay Knopf : Erzeugt eine Roboscript Zeile, die den Programmablauf um die im darunterliegenden Feld ausgewählten Betrag verzögert.
	Goto Knopf: Erzeugt eine RoboScript Zeile, die einen Sprung zu der in "Line" angegeben Roboscript Programmzeile auslöst.
	PTO ON :Schaltet das Point-to-Point verhalten ein
	PTP OFF : Schaltet das Point-to-Point verhalten aus
	Speed: Erzeugt eine RoboScript Zeile, die die Drehgeschwindigkeit der Servos durch den unten bei Level gewählten Wert festlegt. Je höher der Wert ist, desto schneller arbeiten die Servos.
	Insert Move: Erzeugt eine RoboScript Zeile, mit den aktuell über die Schieberegler eingestellen Servopositionen

Copy Copy Knopf: Die Copy Funktion überträgt das gesamte Roboscript Programm in die Zwichenablage des Betriebssystems. Von dort aus kann es mittels "Einfügen" oder "Strg+v" in ein anders Programm eingefügt werden. Durch diese Funktion kann man ein unter RoboScript erstelltes Programm auch in RoboBASIC einfügen.

	Run: Führt das aktuelle RoboScript Programm aus.
	Stop: Beendet die Ausführung des aktuellen RoboScript Programms.

Nutzung von RoboRemocon V2.5

Information

RoboRemocon erlaubt die drahtlose Steuerung des Roboters. Ein mit Roboscript erstelltes Programm wird mit Hilfe von RoboRemocon den Steuertasten einer Fernbedienung zugeordnet. Zu Testzwecken kann RoboRemocon auch ohne vorhandene Fernbedienung genutzt werden.

Diese Anleitung bezieht sich auf RoboRemocon v 2.5. An Software und Handbuch können zur Verbesserung ohne Bekanntgabe Änderungen vorgenommen werden.

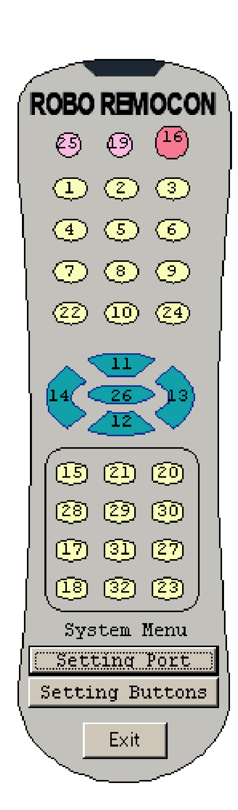
RoboScript ist eine registrierte Software. Erstellen von Kopien, Veröffentlichung, Versand oder Vertrieb dieser Anleitung oder der Software ohne Genehmigung ist untersagt.

Installation

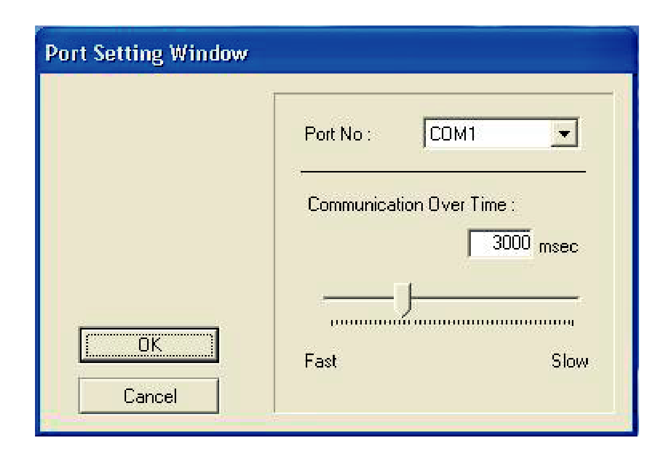
RoboRemocon wird bei der Installation von RoboBASIC ab der Version 2.5 automatisch mit installiert.

Einrichtung von RoboRemocon

- Schließen Sie das Datenkabel an eine serielle Schnittstelle Ihres PCs an
- Verbinden Sie das Datenkabel mit dem Controller
- Schalten Sie den Controller ein.
- Starten Sie RoboRemocon durch Klick auf das entsprechende Icon auf Ihrem Desktop



- Setting Port Klicken sie auf <Setting Port>

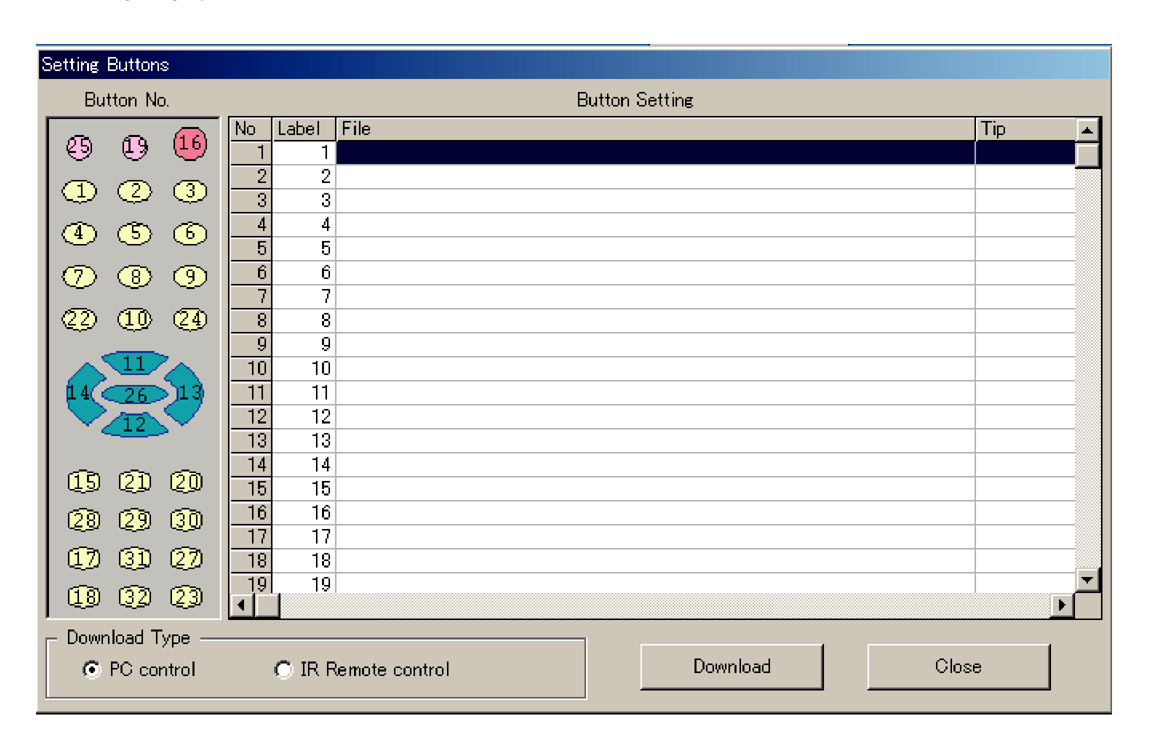


- Wählen Sie unter Port No: die Nummer der seriellen Schnittstelle aus, an dem Ihr Datenkabel angeschlossen ist.

Programmieren mit RoboRemocon

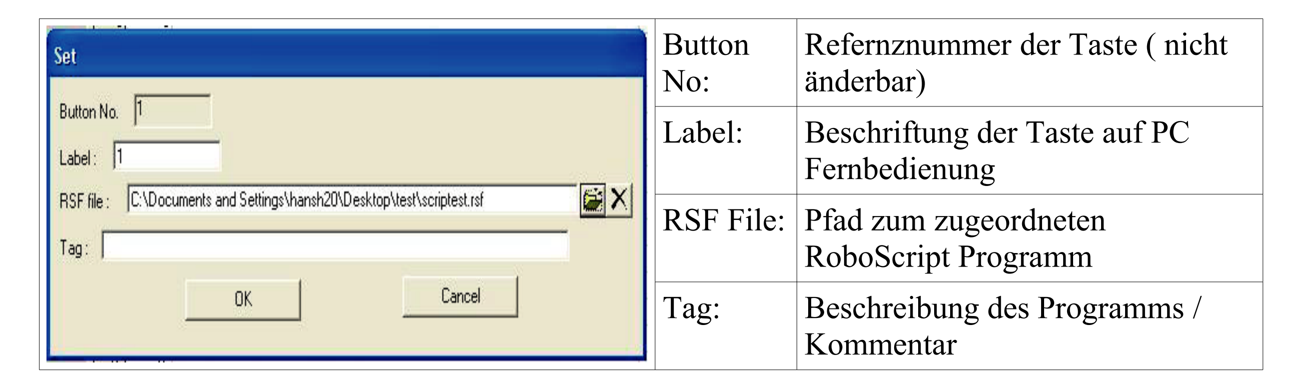
- Wenn noch nicht geschehen, starten Sie Roboremocon durch Klick auf das entsprechende ICON auf Ihrem Desktop und stellen die Verbinden PC - Controller mittels Datenkabel her.

- Setting Buttons : Ein Klick auf diesen Knopf öffnet das "Setting Buttons" Fenster, in dem, den Tasten der Fernbedienung, einzele Roboscript Programme zugeordnet werden können.



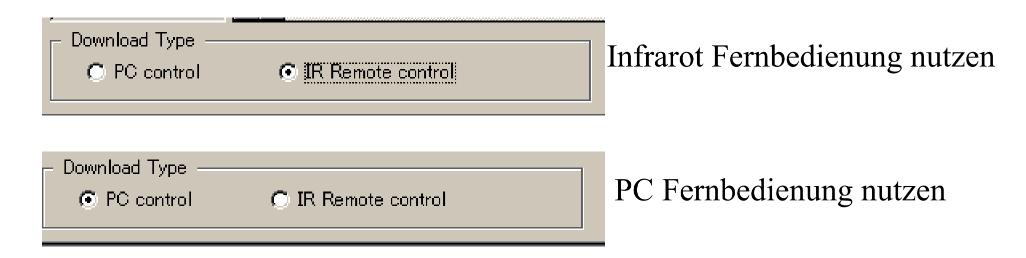
- Klicken Sie mit der Maus entweder, die zur Änderung gewünschte, symbolische Taste auf

der Fernbedienung oder die entsprechende Zeile im Textbereich an, um das "Set" Fenster aufzurufen.



Button No:	Refernznummer der Taste (nicht änderbar)
Label:	Beschriftung der Taste auf PC Fernbedienung
RSF File:	Pfad zum zugeordneten RoboScript Programm
Tag:	Beschreibung des Programms / Kommentar

- Wählen Sie zwischen der vom PC symbolisch dargestellten <PC-Control> oder der IR Fernbedienung <IR Remote control> aus.



Infrarot Fernbedienung nutzen

PC Fernbedienung nutzen

- Download startet die Übersetzung des RoboRemocon Programms und überträgt das Programm zum Controller . Hinweis: Sollten Sie bei der Datenübertragung eine Fehlermeldung erhalten, überprüfen Sie bitte noch einmal die richtige Einstellung der PC-Schnittstelle im <Setting Port> Programm. Beachten Sie auch die Hinweise im der RoboScript Anleitung zur Fehlersuche.
- Exit Exit: Beendet RoboRemocon

Nutzung von RoboRemocon unter RoboBASIC

Es besteht die Möglichkeit, die symbolische Fernbedienung von RoboRemocon in einem RoboBASIC Programm auszuwerten:

Grundsätzliche Vorgehensweise

- Initialisieren Sie mittels DIM-Befehl eine Variable, in der RoboRemocon die Tasteninformation übergeben soll. **Wichtig: Dieses muss die erste initialisierte Variabel sein!**
- Die Übergangsvariabel liefert "0" wenn kein Wert von RoboRemocon vorligt.

.

```
'Demolisting Nutzung von RoboRemocon unter RoboBASIC
'
'
'==== Initalisierung ====
'
DIM RR AS BYTE ' IN "RR" übergibt RoboRemocon die gedrückte
Taste
RR=0       ' RR = 0 gibt Nutzung von Remocon frei

'==== Hauptprogramm =====

Hauptprogramm:

' Wenn keine Taste gedrückt RR=0 -> gehe zu Hauptprogrmm
' Taste 1 gedrückt RR=1  -> Rufe Unterprogramm Knopf1 auf ..
' Taste 2 gedrückt RR=2  -> Rufe Unterprogramm Knopf2 auf ..
                          ' etc...

      ON RR GOTO Hauptprogramm,Knopf1,Knopf2,Knopf3


GOTO Hauptprogramm

' === Unterprogramme ====

Knopf1:
               PRINT "Unterprogramm Knopf1"
GOTO Hauptprogramm  'RETURN

Knopf2:
               PRINT "Unterprogramm Knopf2"
GOTO Hauptprogramm  'RETURN


Knopf3:
               PRINT "Unterprogramm Knopf3"
GOTO Hauptprogramm  'RETURN
```

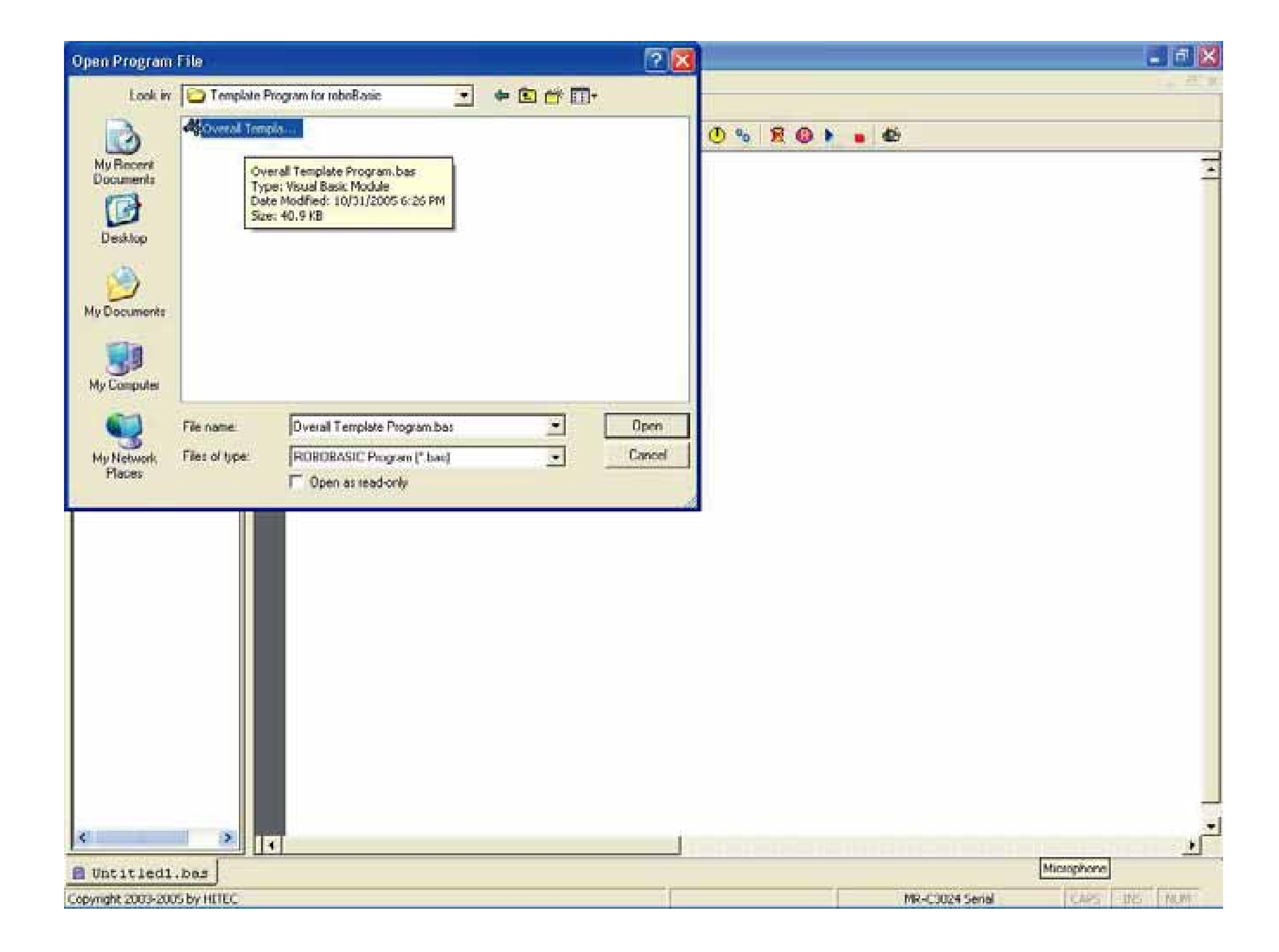
Wichtig: Die Übergabe des Tastenwertes erfolgt ausschließlich über die ERSTE im Programm dimensionierte Variable! Wichtig ist also nicht der Name, sondern dass die gewünschte Variable als erste deklariert wird!

Praktische Anwendungen / Erweiterung der Programmvorlagen

Im Nachfolgenden erhalten Sie eine Einführung in das, dem ROBONOVA beiliegende, umfangreiche RoboBASIC Beispielprogramm, welches sich gut als Basis für eigene Programmprojekte eignet.

Bitte beachten Sie, dass die in der Anleitung zu Grunde liegenden Beispiele, sich von der Programmversion der aktuell ausgelieferten CD im Detail unterscheiden können.

- Starten Sie RoboBASIC und stellen eine Datenverbindung zum Roboter her
- Laden Sie das Musterprogramm "Overall Template.bas" in den RoboBASIC Editor



Grundstruktur des Programms

Das Programm kann in 3 Teile unterteilt werden:

1. Initialisierungs Teil
2. Main - Das Hauptprogramm
3. Unterprogramme für jede Bewegung

Der Initialisierungsteil

```
0002 '== auto_main ===================================
0003 GOTO AUTO
0004 FILL 255,10000
0005
0006
0007 DIM RR AS BYTE
0008 DIM A AS BYTE
0009 DIM A16 AS BYTE
0010 DIM A26 AS BYTE
0011
0012 CONST ID = 0      ' 0:0, 1:32, 2:64, 3:96,
0013
0014 RR = 0
0015
0016 PTP SETON
0017 PTP ALLON
0018
0019 '== motor diretion setting ======================
0020 DIR G6A,1,0,0,1,0,0
0021 DIR G6B,1,1,1,1,1,1
0022 DIR G6C,0,0,0,0,0,0
0023 DIR G6D,0,1,1,0,1,0
0024
0025 '== motor start position read ===================
0026 TEMPO 230
0027 MUSIC "CDE"
0028 GETMOTORSET G24,1,1,1,1,1,0,1,1,1,0,0,0,1,1,1,0,0,0,1,1,1,1,1,0
0029 '== motor power on  ============================
0030 SPEED 5
0031 MOTOR G24
0032 GOSUB standard_pose
0033 '===============================================
0034 MAIN:
0035 'GOSUB robot_voltage
0036 GOSUB tilt_proc
0037
0038 IF RR > 50 THEN GOTO action_proc
0039 '-----------------------------
0040 IF RR = 0 THEN GOTO MAIN1
0041
0042 ON RR GOTO MAIN,K1,K2,K3,K4,K5,K6,K7,K8,K9,K10,K11,K12,K13,K14,K15,K16,K17,K18,K19,K20,K21,K22,K23
0043 GOTO main_exit
```

Erläuterung

' Templates laden
GOTO AUTO ' Ruft Programm zum automatischen Laden der Vorlagen auf
FILL 255,10000 ' Speicher Vorlagen Programm ab Adressbereich 10000 im MR-C3024 controller'

' Variable deklarieren
DIM RR AS BYTE ' Die erste deklariete Variabel wird von RoboRemocon als Übergabevariable genutzt'

' Konstanten deklarieren
CONST ID =0 ' ID für den Kennzeichnung des Remocon Controllers

' Setzen von Grundbedingungen für Servos

Das Hauptprogramm "MAIN"

Der Programmabschnitt "Main" bezeichnet den zentralen Steuerteil des Musterprogrammes. Dieser Programmteil wertet die Benutzereingaben aus und verzweigt zu den entsprechenden Unterprogrammen. Nach abgeschlossenem Unterprogramm springt dieser (meist) wieder zum Hauptprogramm zurück.

```
'================================================
MAIN:
'GOSUB robot_voltage
GOSUB tilt_proc

IF RR > 50 THEN GOTO action_proc
'-----------------------------
IF RR = 0 THEN GOTO MAIN1

ON RR GOTO MAIN,K1,K2,K3,K4,K5,K6,K7,K8,K9,K10,K11,K12,K13,K14,K15,K16,K17,K18,K19,K20,K21,
GOTO main_exit
'-----------------------------
MAIN1:
A = REMOCON(1)
A = A - ID
ON A GOTO MAIN,K1,K2,K3,K4,K5,K6,K7,K8,K9,K10,K11,K12,K13,K14,K15,K16,K17,K18,K19,K20,K21,K
GOTO MAIN
'-------------------------------------------------
action_proc:
A = RR - 50
ON A GOTO MAIN,K1,K2,K3,K4,K5,K6,K7,K8,K9,K10,K11,K12,K13,K14,K15,K16,K17,K18,K19,K20,K21,K
RETURN
'-----------------------------
```

Abbildung 18: Typisches Hauptprogramm in RoboBASIC

Erläuterung

GOSUB robot_voltage ' Aufruf Unterprogramm zur Spannungsüberwachung , kann wahlweise ' auskommentiert sein

GOSUB tilt_proc ' Unterprogramm zur Überprüfung, ob Roboter umgefallen ist, setzt ' vorhandensein des "Tilt" Sensors (Lagesensor) voraus.

IF RR > 50 THEN GOTO action_Proc ' Werte > 50 von Remocon werden als "Action code" zum ' Aufrufen der Templates genutzt, dieser Aufruf geschieht ' in der mit action_Proc: bezeichneten Zeile

IF RR = 0 THEN GOTO MAIN1 ' RR= 0 bedeutet keine Werte von RoboRemocon zugewiesen, Aufruf von MAIN1 ' wertet Befehle der Infrarot steurung aus

' Auswertung des von RoboRemocon gelieferten Wertes RR > 0

ON RR GOTO MAIN,K1,K2,K3,K4,K5,K6,K7,K8,K9,K10,K11,K12,K13,K14,K15,K16,K17,K18,K19,K20,K21,K22,K23,K24,K25,K26,K27,K28,K29,K30,K31,K32

GOTO main_exit ' Rücksprung zu MAIN: , Schleife Hauptprogramm

```
MAIN1:              ' Auswertung der Infrarot Steuerung
A = REMOCON(1)      ' Lese Wert der Infrarot Bedienung in A ein
A = A - ID          '  Wert A für benutzte Remocon ID berichtigen


' In Abhängigkeit von A entsprechendes Unterprogramm aufrufen
ON A GOTO
MAIN,K1,K2,K3,K4,K5,K6,K7,K8,K9,K10,K11,K12,K13,K14,K15,K16,K17,K18,K19,K20,K21,
K22,K23,K24,K25,K26,K27,K28,K29,K30,K31,K32

GOTO MAIN           ' Rücksprung zu MAIN: Schleife Hauptprogramm
```

Unterprogramme für Bewegungsmuster der Fernbedienung

Im Hauptprogramm wird in Abhängigkeit von dem der Fernbedienung gelieferten Wert zu den einzelnen Sprungmarken k1: bis k32: verzweigt.

```
0064 '==================================================
0065 k1:
0066     GOSUB bow_pose
0067     GOSUB standard_pose
0068     GOTO main_exit
0069 k2:
0070     GOSUB hans_up
0071     DELAY 500
0072     GOSUB standard_pose
0073     GOTO main_exit
0074 k3:
0075     GOSUB sit_down_pose
0076     DELAY 1000
0077     GOSUB standard_pose
0078     GOTO main_exit
0079 k4:
0080     GOSUB sit_hans_up
0081     DELAY 1000
0082     GOSUB standard_pose
0083     GOTO main_exit
0084 k5:
0085     GOSUB foot_up
0086     GOSUB standard_pose
0087     GOTO main_exit
0088 k6:
0089     GOSUB body_move
0090     GOSUB standard_pose
0091     GOTO main_exit
0092 k7:
0093     GOSUB wing_move
0094     GOSUB standard_pose
0095     GOTO main_exit
0096 k8:
0097     GOSUB right_shoot
0098     GOSUB standard_pose
0099     DELAY 500
0100     GOSUB left_shoot
0101     GOSUB standard_pose
```

Abbildung 19: Bereich der für die einzelnen den Tasten der Fernbedienung zugeordneten Befehlen

Der Aufbau kann auf ein Grundmuster reduziert werden:

```
k1:                          ' Sprungmarke
GOSUB Bewegung               ' Grundbewegung aufrufen
...                          ' sonstige Befehle abarbeiten
...
GOTO main_exit               ' Springe zur Sprungmarke  main_exit (hier erfolgt dann
                             ' Rücksprung ins Hauptprogramm)
```

Neue Grundbewegung / Funktionen der Fernbedienung zuordnen

Sie können das mitgelieferte Musterprogramm durch eigene Anwendungen erweitern.
Nehmen wir an, Sie möchten über die Fernbedienung Taste "E" die blaue LED im Kopf des ROBONOVA blinken lassen.

Zuerst wird ein Programm benötigt, welches die LED wie gewünscht ein- und ausschaltet:

```
''lässt ROBONOVA LED blinken
LED_Blinken:
     OUT 52,1
     DELAY 1000
     OUT 52,0
     DELAY 1000
RETURN
```

Als nächstes ermitteln Sie aus der Tabelle der Fernbedienung den Tastencode für die "E" Taste:

REMOCON key assignment table

ACTION	key	Motion	Variable	Code
0	power	ON : motor on→Basic Position OFF: Sitting Position→ motor off	A16	16
1	1	Bow → Basic Position		1
2	2	Raise arms → Basic Position		2
3	3	Sit → Basic Position		3
4	4	Sit → Raise arms → Basic Position		4
5	5	Raise a leg → Basic Position		5
6	6	Spread the legs → Extend arms → Right↔left tilt → Basic Position		6
7	7	Flap arms like a bird		7
8	8	Kick		8
9	9	Handstand		9
10	0	Walk fast		10
11	*	Left turnabout		22
12	#	Right turnabout		24
13	▲	Forward		11
14	◀	Left move		14
15	■	Sit↔Stand up	A26	26
16	▶	Right move		13
17	▼	Reverse		12
18	△	Front tumbling		21
19	◁	Left cartwheel		28
20	□	Front attack		29
21	▷	Right cartwheel		30
22	▽	Rear tumbling		31
23	A	Left attack		15
24	B	Right attack		20
25	C	Left front jab		17
26	D	Right front jab		27
	E,F,G	27,28,29 Spare 18,32,23		18

Abbildung 20: Zuordnung Infrarot Fernbedienungstaste - Tastencode

Dieser ergibt sich aus der letzten Zeile der Tabelle mit "18".

Falls noch nicht geschehen, starten Sie nun RoboBASIC und stellen die Datenverbindung zu ROBONOVA her.

Laden Sie das Musterprogram "Overall Tempate Program.bas" und speichern dieses unter einem neuen Namen ab, um so eine Arbeitskopie zu erstellen.

Suchen Sie mittels der "Find" Funktion des Editors im Quelltext die Textmarke "k18:"

```
k18:                                    ' E
        TEMPO 230
        MUSIC "C"
        GOTO main_exit
```

Text 1: Label k18: in Quellfile vor der Änderung

Die Marke k18: wird vom Hauptprogramm aus angesprungen, wenn die Taste "E" auf der Fernbedienung gedrückt wird. Im Musterlisting wird ein Ton ausgegeben, dieses soll durch ein Blinken der LED ersetzt werden.

Ändern Sie die betreffenden Zeile nun so, dass das Unterprogramm "LED_Blinken" stattdessen aufgerufen wird:

```
k18:
        GOSUB LED_Blinken
        GOTO main_exit
```

Nun muss noch der neue Programmteil "LED_Blinken:" in das Programm eingefügt werden

Diese kann im Prinzip direkt im Anschluss den geänderten Teil erfolgen, es hat sich aber als sinnvoll erwiesen, die Programmteile mit eigenen Erweiterungen zentral an einer Stelle im Quelltext zu halten.

Der vorgeschlagene Platz befindet sich direkt hinter der für die Fernbedienung zuständigen Routinen, hier ist zuerst das Programm zur Batteriespannungsüberwachung "robot_voltage:"

```
'================================================
robot_voltage:                    ' [ 10 x Value / 256 = Voltage]
    DIM v AS BYTE

    A = AD(6)

    IF A < 148 THEN               ' 5.8v

    FOR v = 0 TO 2
    OUT 52,1
    DELAY 200
    OUT 52,0
    DELAY 200
    NEXT v

    RETURN
'
```

Abbildung 21: Empfohlende Stelle, um eigene Unterprogramme in den vorhanden Musterquelltext einzufügen

Der nachfolgende Auszug aus dem geänderten Quelltext zeigt eine mögliche Vorgehensweise:

```
k32:                    ' G
     TEMPO 230
     MUSIC "E"
     GOTO main_exit
'================================================
' lässt ROBONOVA LED blinken
LED_Blinken:
     OUT 52,1
     DELAY 1000
     OUT 52,0
     DELAY 1000
RETURN

robot_voltage:                           ' [ 10 x Value / 256 = 
Voltage]
     DIM v AS BYTE
     A = AD(6)
```

Text 2: Geänderter Quelltext erweitert um die Funktion LED_Blinken

Nach Änderung des Quelltextes muss dieser noch Übersetzt und zum Roboter übertragen werden: <F9 Taste>

Nach dem erfolgreichem Upload steht nun die neue Funktion "Blinken" auf der "E-Taste" der Fernbedienung zur Verfügung.

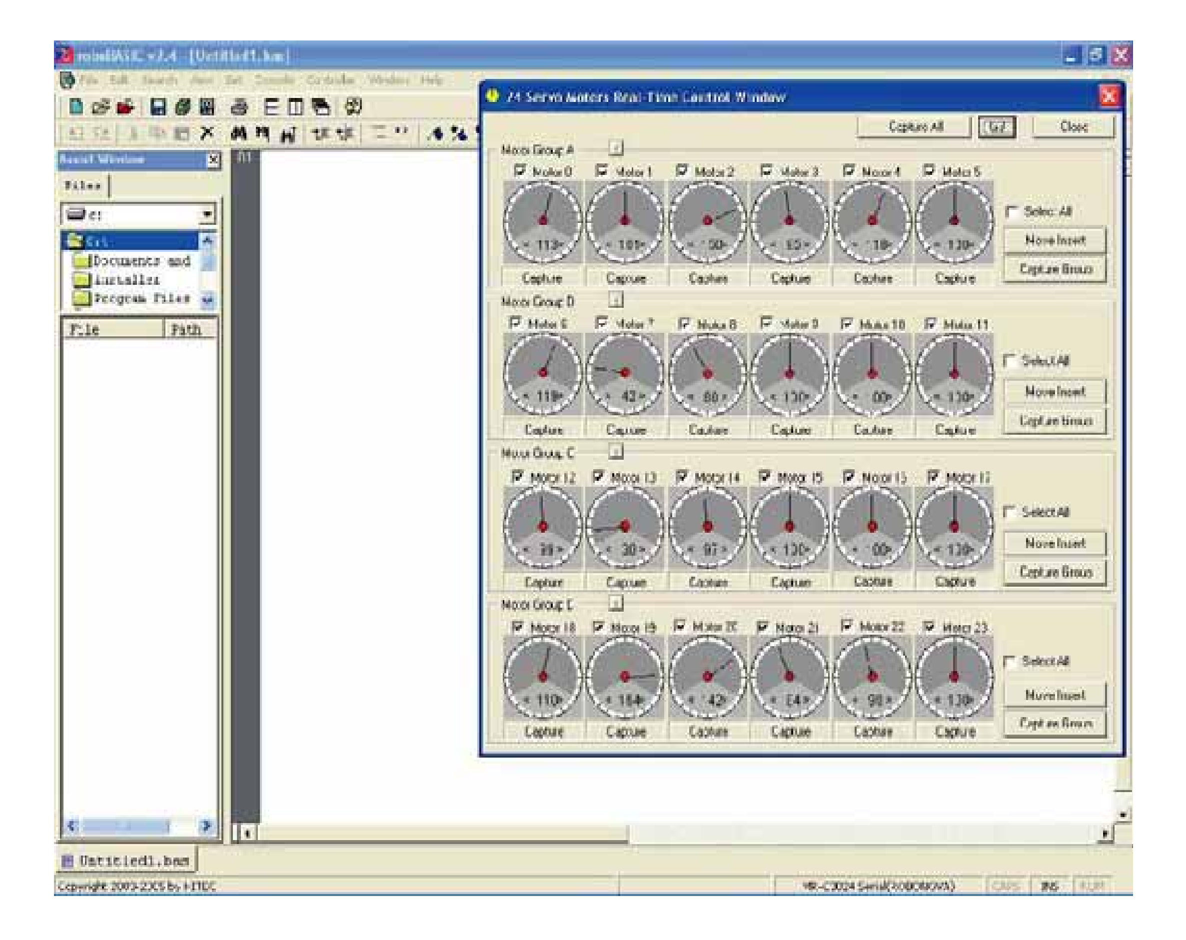
Eigene Programme erstellen

Nutzung der Motion Feedback Funktionen

Motion Feedback steht für die Fähigkeit der im ROBONOVA zur Anwendung kommenden Servos, eine manuelle Einstellung der Servoposition zu erlauben und diese dann wieder mittels eines Programms auslesen zu können.

- Starten Sie die RoboBASIC Programmierumgebung und verbinden Sie PC und Controller mit dem Datenkabel

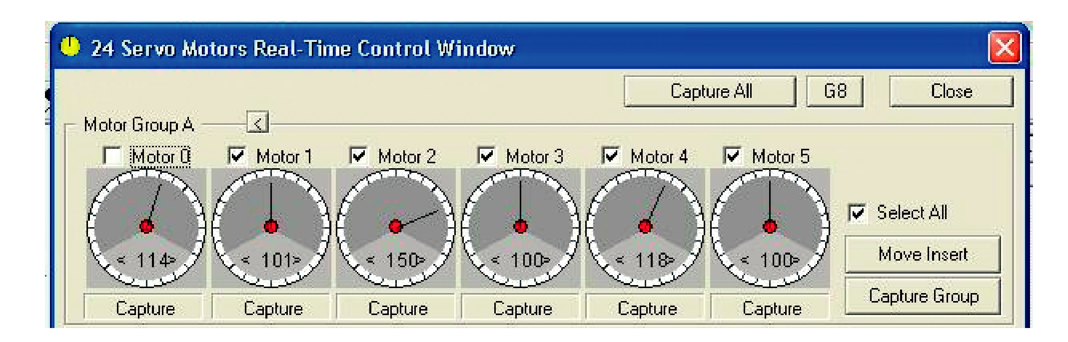
- Positionieren Sie den Cursor des Editors auf die Stelle im Quelltext, wo die Servobewegungen(MOVE -Befehl) eingefügt werden sollen
  - Start der Echtzeit Servokontrolle durch Klick auf das Symbol in der Symbolleiste
  - alternativ: Aufruf über das Menü <Controller> Unterpunkt <Servo motor realtime control>
  - alternativ: Aufruf über F7 Taste (Shortcut)



Manuelle Einstellung der Servopositionen:

Hier können sie alle zuvor auf Freilauf geschalteten Servos von Hand in die Wunschposition bewegen, dieses Verfahren ist bei Robotern sehr effektiv in der Anwendung, da bei den meisten Bewegungen mehr als ein Servo beteiligt ist.

Vorgehensweise



- Schalten Sie die, für die Bewegung, benötigten Servos durch klick auf das Markierungsfeld auf Freilauf (der Motor des Servos wird hier stromlos geschaltet) Im obigen Beispiel ist der Servo Motor 0 im Freilaufmodus.
- Bewegen Sie die, im Freilaufmodus befindlichen, Servos in die gewünschten Positionen
- Klicken Sie den Capture Knopf an, um die neue Servoposition einzulesen
- Klicken Sie erneut in das Markierfeld des im Freilauf befindlichen Servos, der Servomotor wird wieder mit Strom versorgt und hält nun aktiv die neue Position.
- Um diese Servostellungen in Ihren Quelltext einzuzufügen, betätigen sie den <MOVE INSERT> Knopf. Hinweis: In der erzeugten MOVE Befehlszeile werden nur die Servos aus der jeweiligen Gruppe berücksichtigt, die als aktiv (Servo hält Position) gekennzeichnet sind. Alle Servos im Freilauf werden mit Lehrzeichen anstelle der Position zurückgeliefert.

Einstellung der Servoposition mittels Maus oder Pfeiltasten

Warnung! Bei direkter Einstellung der Servos können die Servos unter Umständen durch versehentliches Verstellen sehr heftige Bewegungen ausführen

- Aktivieren Sie den gewünschten Servomotor Motor 0 durch setzen des Markierfeldes
- Klicken Sie mit der Maus in die Positionsanzeige
- Bei gedrückter rechter Maustaste können Sie den Zeiger auf die gewünschte neue Position stellen, nach loslassen der Maustaste nimmt das Servo sofort die neue Position ein.
- Nutzen sie die Pfeil links / Pfeil rechts Tasten um die Servoposition in kleinen Schritten zu ändern.
- Um diese Servostellungen in Ihren Quelltext einzuzufügen, betätigen sie den <MOVE INSERT> Knopf. Hinweis: In der erzeugten MOVE Befehlszeile werden nur die Servos aus der jeweiligen Gruppe berücksichtigt, die als aktiv (Servo hält Position) gekennzeichnet sind. Alle Servos im Freilauf werden mit Leerzeichen anstelle der Position zurückgeliefert.

Tipp: Nutzen Sie das <Select all> Markierfeld, wenn sie alle Servos einer Gruppe gleichzeitig ändern möchten.

Abbildung 22: Editor nach Einfügen einer Servoposition

Um dieses Programm ausführen zu können, müssen noch einige einfache Initialisierungen eingefügt werden:



```
'Init Settings
GETMOTORSET G24,1,1,1,1,1,0,1,1,1,0,0,0,1,1,1,0,0,0,1,1,1,1,1,0
SPEED 5
MOTOR G24
```

Text 3: Typische Servo Initialisierung für ROBONOVA

Info: GETMOTORSET G24 ..... ' Sorgt für eine definierte, unkritische Startposition

Speed 5 ' legt die Servogeschwindigkeit auf Stufe 5 fest

MOTOR G24 ' Servogruppe 24 wird benutzt

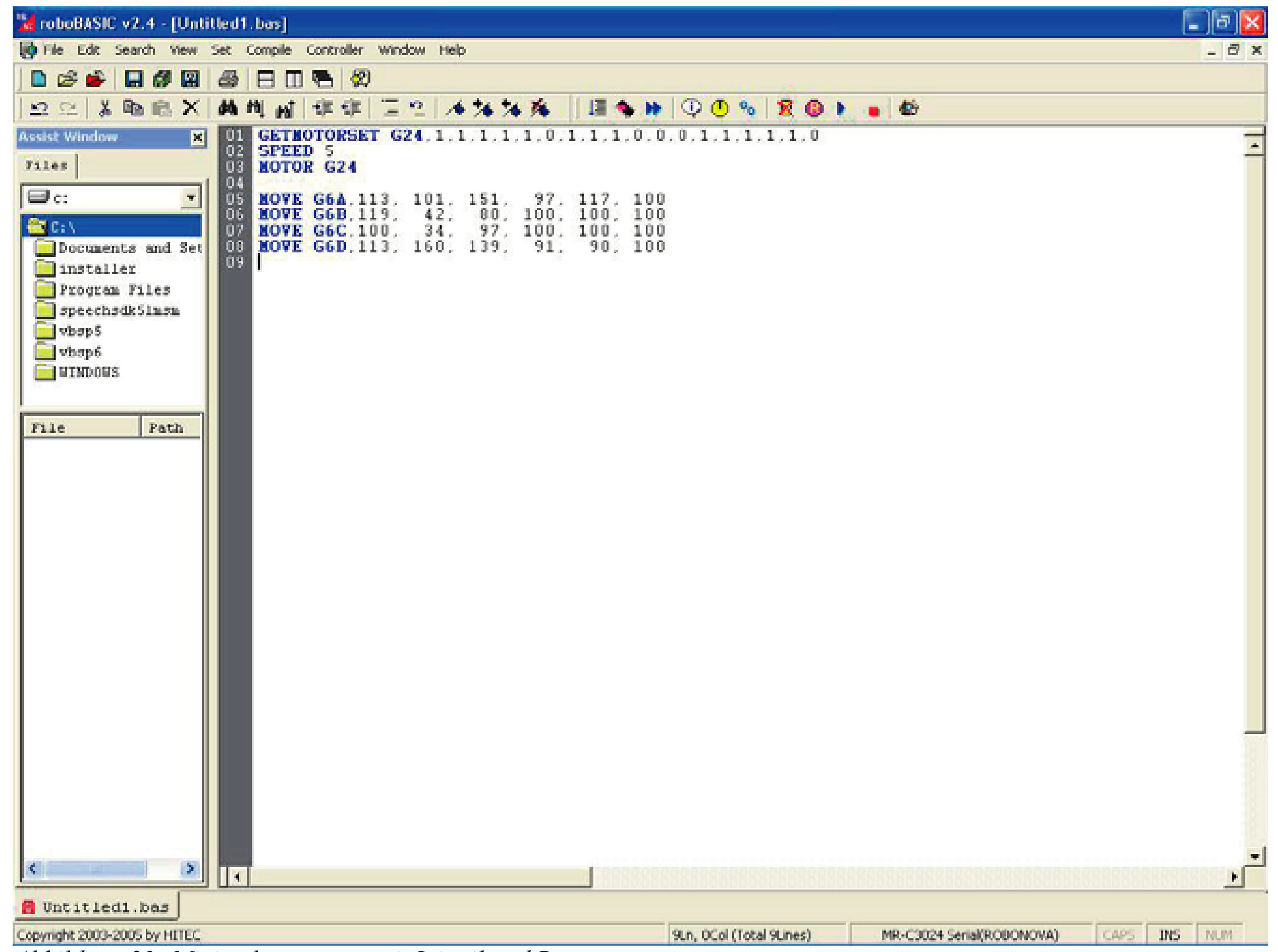


Abbildung 23: Minimalprogramm mit Initteil und Bewegung

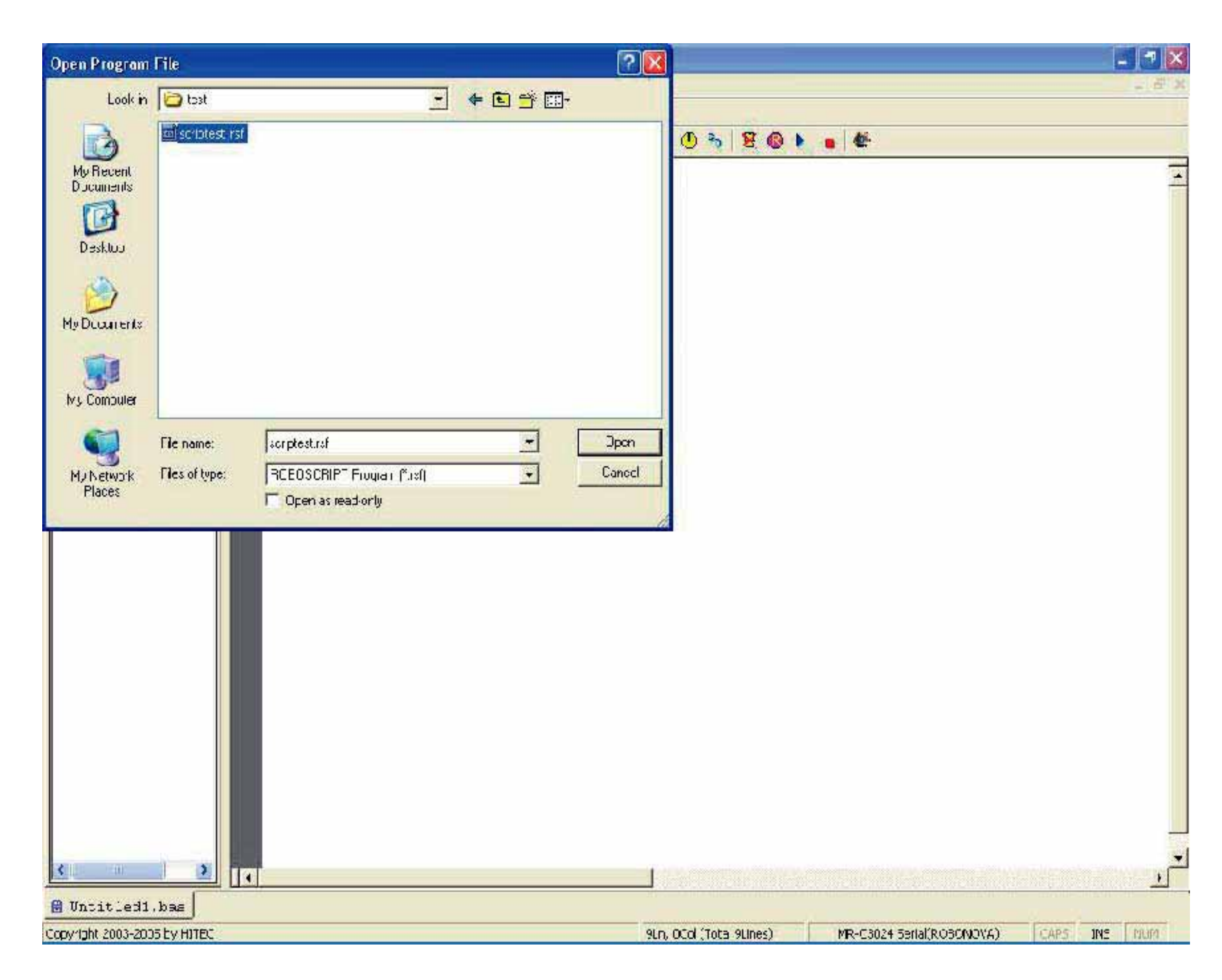
Das neu erstellte Programm kann dann mittels <F9> Taste übersetzt und zum Roboter übertragen werden.

Einfügen von RoboScript Files in RoboBasic

Um mit einem eigenen RoboBasic Programm zu beginnen, gibt es auch die Möglichkeit ein schon in RoboScript erstelltes Bewegungsmuster als Grundlage für Erweiterungen in RoboBasic zu übernehmen.

Vorgehensweise:

- Starten Sie die RoboBASIC Programmierumgebung und verbinden Sie PC und Controller mit dem Datenkabel
- Positionieren Sie den Cursor des Editors auf die Stelle im Quelltext, wo die Servobewegungen(MOVE -Befehl) eingefügt werden sollen
- Öffenen Sie das Dateiauswahlauswahlfenster zum Laden eines Programms im Editor <CNTRL-O>
- Ändern Sie den Dateityp von "Robobasic Program (.bas)" auf „ROBOSCRIPT Program(.rsf)" um RoboScript Dateien in der Anzeige zu filtern



- Laden Sie das gewünschte RoboScript File in den Editor

Um aus diesem RoboScript ein lauffähiges RoboBasicprogramm zu erstellen, fügen Sie die nachfolgenden Initialisierungszeilen zu Beginn des neuen Programms ein:

```
'Init Settings
GETMOTORSET G24,1,1,1,1,1,0,1,1,1,0,0,0,1,1,1,0,0,0,1,1,1,1,1,0
SPEED 5
MOTOR G24
```

Text 4: 3: Typische Servo Initialisierung für ROBONOVA

<u>**Info:**</u> GETMOTORSET G24 ..... ' Sorgt für eine definierte, unkritische Startposition
Speed 5 ' legt die Servogeschwindigkeit auf Stufe 5 fest
MOTOR G24 ' Servogruppe 24 wird benutzt

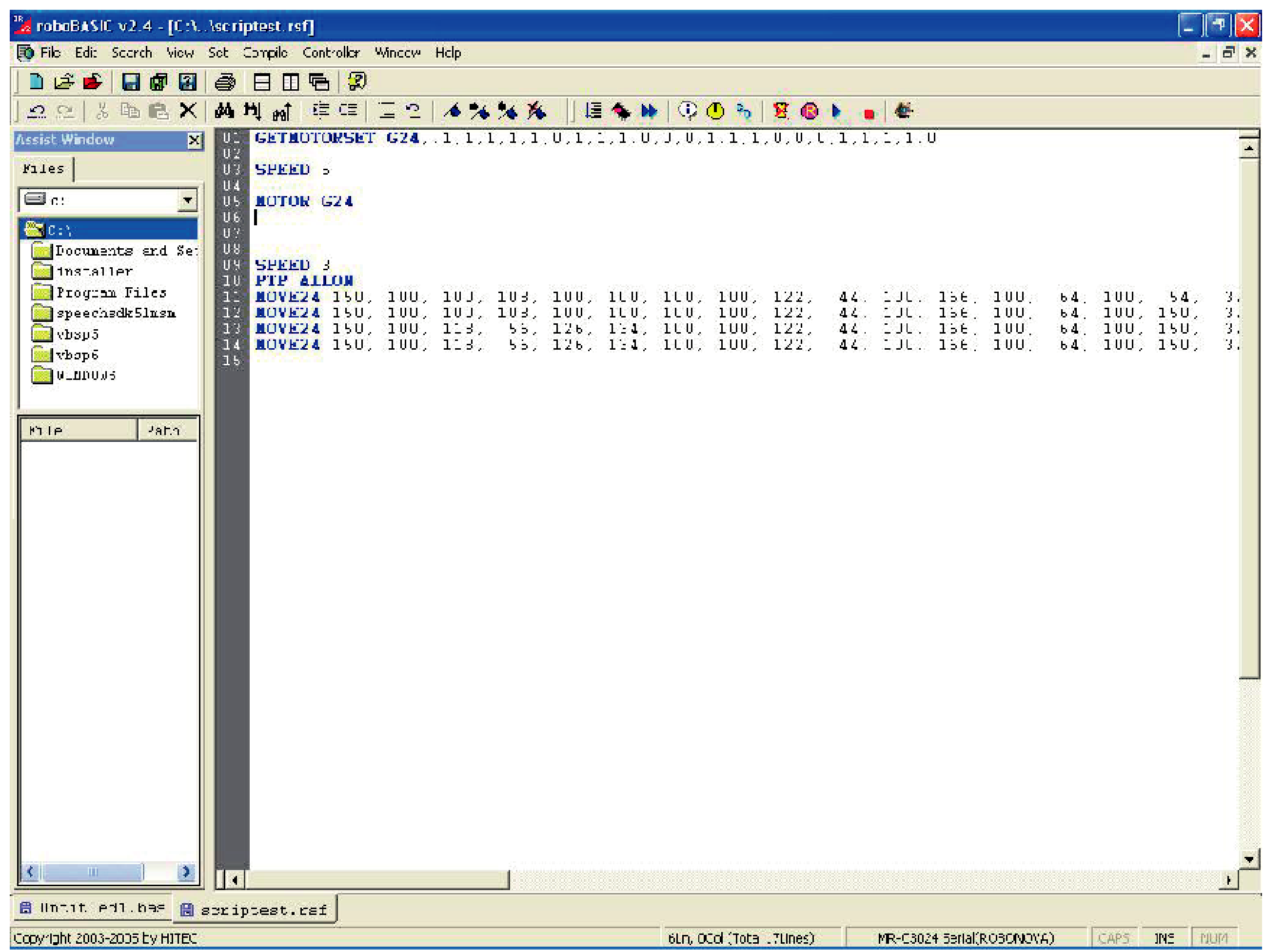


Abbildung 24: Importiertes RoboScript File ergänzt im Initalisierung der Servos

Das neu erstellte Programm kann dann mittels <F9> Taste übersetzt und zum Roboter übertragen

werden.

Gleichzeitiger Betrieb mehrerer Roboter mit Infrarot Fernbedienung

Bei der Nutzung der ROBONOVA Infrarot Fernbedienung REMONCON ist es möglich, bis zu vier Roboter gleichzeitig mit vier Fernbedienungen unabhängig voneinander störungsfrei zu steuern.

Dazu wird jeder Fernbedienung ein eigener Codebereich zugewiesen, der ausschließlich von dieser Fernbedienung genutzt wird. Entsprechend ist die Auswertung im Empfangsprogramm an den Codebereich der Fernbedienung vorzunehmen.

Schnelleinstieg

- Laden Sie das Programm "Overall Template Program.bas" in den RoboBASIC Editor
- Passen Sie die Zuweisung für die Konstante „ID“ in der Zeile

```
CONST ID = 0     ' 1:0, 2:32, 3:64, 4:96
```

  gemäß der Tabelle an und übertragen das so geänderte Programm zum Roboter.

- Stellen Sie die REMOCON Fernbedienung gemäß der Tabelle passend zur gewählten Konstanten „ID“ um: Drücken und halten Sie hierzu die <PF1> Taste der Fernbedienung zusammen mit der Ziffertaste [1..4] für 2 Sekunden gedrückt.

Wertezuweisung Konstante ID im Quelltext	***Tastenkombination REMOCON Fernbedienung***
CONST ID = 0	*<PF1> + <1> für 2 Sekunden gedrückt halten*
CONST ID = 32	<PF1> + <2> *für 2 Sekunden gedrückt halten*
CONST ID = 64	<PF1> + <3> *für 2 Sekunden gedrückt halten*
CONST ID = 96	<PF1> + <4> *für 2 Sekunden gedrückt halten*

Hinweis: Nach Austausch der Batterien der Fernbedienung wird die Fernbedienung auf Werkseinstellung zurückgesetzt und muss ggf. neu angepasst werden!

Erläuterung

Für jede Fernbedienung steht ein Coderaum von 31 Befehlen zur Verfügung: (vgl. auch Tastencode im Anhang Abbildung 27 Seite 54)

REMOCON ID Nummer	***Key Code Bereich***
1	0..31
2	32..63
3	64..95
4	96..127

Nutzung mehrerer Fernbedienung für einen Roboter

Auch ist es möglich - falls der Befehlsvorrat einer REMOCON Fernbedienung nicht ausreichend ist- bis zur vier Fernbedienungen für einen ROBONOVA einzusetzen.

Umstellen des REMOCON Codebereiches

- Drücken Sie sie PF1 Taste auf der REMOCON Fernbedienung und halten diese gedrückt.
- Drücken Sie gleichzeitig zur PF1 Taste ein Ziffertaste 1..4 entsprechend der gewünschten REMOCON ID Nummer.
- Halten Sie diese Tastenkombination für 2 Sekunden unverändert fest.

WICHTIG: Nach einem Wechsel der Fernbedienungsbatterien wird die ID wieder auf die Werksgrundeinstellung ID=1 zurückgesetzt, die ID ist also ggf. nach Austausch der Fernbedienungsbatterien erneut umzustellen.

Hinweis : Nach erfolgter Umstellung der REMOCON Fernbedienung kann eine Anpassung der Robotersoftware notwendig sein!



Abbildung 25: Abbildung 26: Umstellung der REMOCON ID durch gleichzeitigen Drücken der PF1 Taste und der gewünschten ID

Anpassung der mitgelieferten Beispielsoftware

Die Beispielsoftware ist für eine einfache Anpassung an die benutzte REMONCON ID ausgelegt:

- Starten Sie RoboBASIC und stellen einen Datenverbindung zum Controller her
- Laden Sie das Programm "Overall Template Program.bas" in den Editor

```
'================================================
' templet program
'
' RR : internal parameter variable / ROBOREMOCON / Action command
' A  : temporary variable          / REMOCON
' A16,A26 : temporary variable
'
'== auto_main ======================================
GOTO AUTO
....
CONST ID = 0     ' 1:0, 2:32, 3:64, 4:96,
```

Hier ist die Zeile:

CONST ID = 0 ' 1:0, 2:32, 3:64, 4:96,

entsprechend der gewählten REMONCON ID Nummer wie im Kommentar erwähnt, anzupassen:

REMOCON ID Nummer	***Const ID =***
1	0
2	32
3	64
4	96

Die Funktion der Konstanten ID wird bei einem Blick auf das Programm leicht nachvollziehbar:

```
'-----------------------------
MAIN1:
A = REMOCON(1)
A = A - ID
ON A GOTO
MAIN,K1,K2,K3,K4,K5,K6,K7,K8,K9,K10,K11,K12,K13,K14,K15,K16,K17,K18,K19,K20,
K21,K22,K23,K24,K25,K26,K27,K28,K29,K30,K31,K32
GOTO MAIN
```

Anhang

Nutzung des Hitec Multi Protocol Interfaces (HMI)

Hinweis: RoboBASIC bietet direkte Unterstützung für das HMI Protokoll, die nachfolgenden Informationen richten sich somit nur an erfahrene Anwender, die ROBONOVA mit eigenen alternativen Controllern nutzen möchten.

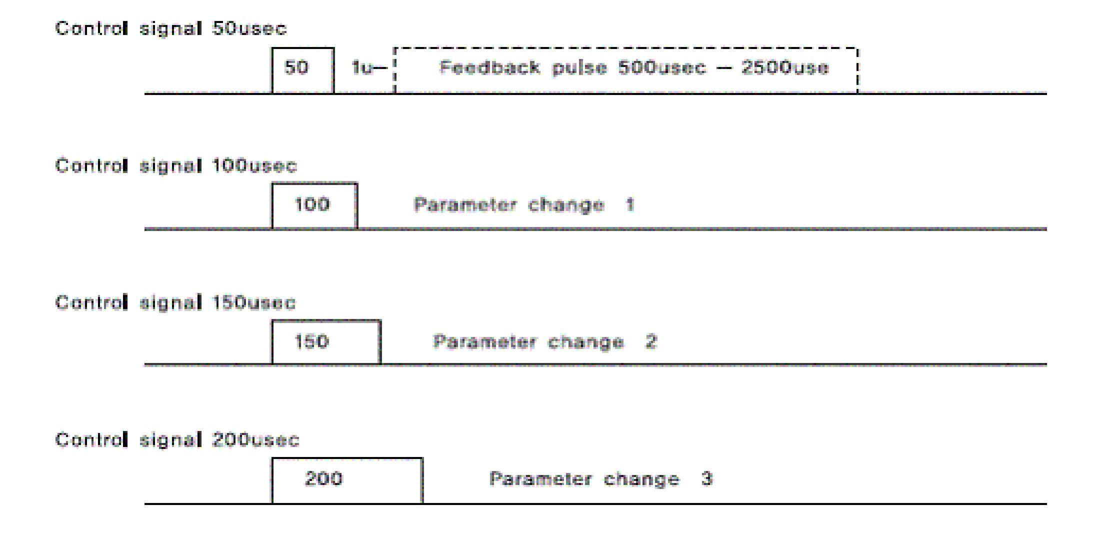
Motion Feedback Funktion

Servos, die das HMI Protokoll unterstützen (z.B. HSR-8498HB), ermöglichen die Abfrage der aktuellen Servoposition.

Dazu wird der Port des Controllers, an dem das PWM Signal für die Servoansteuerung erzeugt wird, kurzfristig als Eingansport genutzt, um den Antwortimpuls des Servos auszuwerten.

Das Servo wird eine vorgegebene Bewegung ausführen, wenn 4 Arten von Impulsen von einer äußeren Quelle eingegeben werden.

1) 50usec Impulse Breite / Wert: Position (Winkel) Rückmeldung
2) 100usec Impulse Breite / Wert: Verwendung Servo Parameter 1 (default)
3) 150usec Impulse Breite / Wert: Verwendung Servo Parameter 2
4) 200usec Impulse Breite / Wert: Verwendung Servo Parameter 3



- Schalten Sie den Ausgangsport des Controllers auf Eingangsbetriebsart um (ggf. hochohmigen Pull Up Widerstand nutzen)
- Werten Sie den vom Servo erzeugten Feedback Impuls aus: Die Impulslänge entspricht dabei dem Impuls, der notwendig ist, um das Servo aktiv in die entsprechende Position zu bringen.

  Hinweis: Ein Fehler von bis zu 10 % ist technisch bedingt möglich.

Anpassung der Unterspannungswarnung

Unterschreitet die Batteriespannung des ROBONOVA einen Grenzwert (Vorgabe 5.8 Volt) , so kann die im Kopf des Roboters angebrachte LED durch Blinken die Unterspannung anzeigen. Das dazu notwendige RoboBASIC Programm ist Teil des mitgelieferten Musterprogrammes, eine Anpassung an die individuellen Wünsche ist leicht durchführbar:

- Überprüfen Sie zuerst, ob das Unterprogramm "robot_voltage" im Hauptprogramm nicht auskommentiert ist! Um das Unterprogramm zu nutzen, Kommentarzeichen <'> ggf. entfernen!

Abbildung 26: Listing mit auskommentierter Spannungsüberwachung durch

```
0029 '===============================================
0030 MAIN:
0031 'GOSUB robot_voltage
0032 GOSUB tilt_proc
0033
0034 IF RR > 50 THEN GOTO action_proc
0035 '-------------------------------
0036 IF RR = 0 THEN GOTO MAIN1
0037
0038 ON RR GOTO MAIN,K1,K2,K3,K4,K5,K6,K7,K8,K9,K10,K11,K12,K13,K
0039 GOTO main_exit
0040 '-------------------------------
```

Unterprogramm robo_voltage

- Suchen Sie das Unterprogramm "robot_voltage:" im Quelltext

```
'================================================
robot_voltage:                          ' [ 10 x Value / 256 = Voltage]
    DIM v AS BYTE

    A = AD(6)

    IF A < 148 THEN                     ' 5.8v

    FOR v = 0 TO 2
    OUT 52,1
    DELAY 200
    OUT 52,0
    DELAY 200
    NEXT v

    RETURN
```

Erläuterung:

A = AD(6) ' Der, dem aktuellen Spannungswert proportionale, Wert in A einlesen

IF A < 148 .THEN

... Blinken ....

endif

Return

Der Wert 148 entspricht dabei 5. 8 Volt Grenzspannung, was sich nach der Formel:

$$Wert = \frac{Grenzspannung * 256}{10}$$

ergibt.

entsprechend eingesetzt :

Wert = 5,8 * 256 / 10 = 148,48 -> da RoboBASIC hier ein Byte erwartet, Wert = 148

Beispiel:

Für 6 Volt Grenzspannung ergibt sich somit:

Wert = 6 * 256 / 10 = 153,6 -> Wert = 153

Tastencode Infrarot Fernbedienung

REMOCON key assignment table

ACTION	key	Motion	Variable	Code
0	power	ON : motor on->Basic Position OFF: Sitting Position-> motor off	A16	16
1	1	Bow -> Basic Position		1
2	2	Raise arms -> Basic Position		2
3	3	Sit -> Basic Position		3
4	4	Sit -> Raise arms -> Basic Position		4
5	5	Raise a leg -> Basic Position		5
6	6	Spread the legs -> Extend arms -> Right-left tilt -> Basic Position		6
7	7	Flap arms like a bird		7
8	8	Kick		8
9	9	Handstand		9
10	0	Walk fast		10
11	*	Left turnabout		22
12	#	Right turnabout		24
13	▲	Forward		11
14	◀	Left move		14
15	■	Sit<->Stand up	A26	26
16	▶	Right move		13
17	▼	Reverse		12
18	△	Front tumbling		21
19	◁	Left cartwheel		28
20	□	Front attack		29
21	▷	Right cartwheel		30
22	▽	Rear tumbling		31
23	A	Left attack		15
24	B	Right attack		20
25	C	Left front jab		17
26	D	Right front jab		27
	E,F,G	27,28,29 Spare 18,32,23		18

Abbildung 27: Tastencodetabelle REMOCON Infrarot Fernbedienung im Auslieferungszustand

ROBOBASIC

Befehls-Bedienungsanleitung

v2.10

Eingetragenes Warenzeichen

Windows ist das eingetragene Warenzeichen der Microsoft Corporation. ROBOBASIC ist registrierte Software der miniROBOT inc.

Hinweis

Diese Anleitung erklärt die Befehle, welche in roboBASIC genutzt werden. Hitec ist für keinen Missbrauch verantwortlich. Diese Anleitung kann ohne vorherige Ankündigung geändert werden, um die Leistung des Produktes zu verbessern.

Inhaltsverzeichnis

Kapitel 1

Befehlszusammenfassung

für roboBASIC

Befehlszusammenfassung

RoboBASIC ist eine spezielle Programmiersprache zur Steuerung von Robotern. RoboBASIC stellt eine Erweiterung der allgemeinen Grundprogrammiersprache mit Befehlen zur Steuerung von Robotern dar.

(2) bedeutet, dass dieser Befehl ausschließlich von Controllern der MR-C2000 Serie ausgeführt werden kann
(3) bedeutet, dass dieser Befehl ausschließlich von Controllern der MR-C3000 Serie ausgeführt werden kann

Befehle, die der Deklaration/Definition dienen

DIM	Variable deklarieren
AS	Variable bei der Deklaration als Variable definieren
CONST	Konstante deklarieren
BYTE	Variable bei der Deklaration als Byte definieren
INTEGER	Variable bei der Deklaration als Integer definieren

Ablaufsteuerbefehle

IF	Beginn einer bedingten Anweisung
THEN	Nächste Anweisung ausführen, wenn die Bedingung wahr ist
ELSE	Nächste Anweisung ausführen, wenn die Bedingung falsch ist
ELSEIF	Beginn einer anderen bedingten Anweisung
ENDIF	Ende der bedingten Anweisung
FOR	Beginn einer Wiederholungsanweisung
TO	Zuweisung des Wiederholungsbereichs einer Wiederholungsanweisung
NEXT	Ende einer Wiederholungsanweisung
GOTO	Teilung des Programmablaufs
GOSUB	Aufruf einer untergeordneten Routine
RETURN	Rückkehr zum Programm aus der untergeordneten Routine
END	Ausführung des Programms beenden
STOP	Ausführung des Programms stoppen
RUN	Programm fortlaufend ausführen
WAIT	Warten, bis das Programm vollständig ausgeführt wurde
DELAY	Programmausführung um einen gewählten Zeitraum verzögern

(2)	BREAK	Programmausführung pausieren und in den Fehlerbehebungsmodus wechseln

Digitale Signaleingabe und -ausgabebefehle

	IN	Signal vom Eingangsport lesen
	OUT	Signal zum Ausgangsport senden
	BYTEIN	Byte-Signal vom Eingangsport der Byte-Einheit lesen
	BYTEOUT	Byte-Signal an den Ausgangsport der Byte-Einheit lesen
(2)	INKEY	Eingehender Schlüssel vom Eingangsport
	STATE	Status des Ausgangsport
	PULSE	Impuls-Signal an den Ausgangsport senden
	TOGGLE	Status des Ausgangsports zurücksetzen
(3)	KEYIN	Analoge Tastenblockeingabe empfangen

Befehle für den Speicher

	PEEK	Daten vom Controller-Arbeitsspeicher lesen
	POKE	Daten in den Controller-Arbeitsspeicher schreiben
	ROMPEEK	Daten vom externen EEPROM-RAM des Controllers lesen
	ROMPOKE	Daten in den externen EEPROM-RAM des Controllers lesen

Befehle für das LCD

	LCDINIT	Initialisieren des LCD-Moduls
	CLS	Alle Zeichen im LCD-Modul löschen
	LOCATE	Zeichenplatzierung im LCD-Modul bestimmen
	PRINT	Buchstaben in LCD-Modul anzeigen
	FORMAT	Typ-Format welches auf dem LCD-Modul angezeigt wird einstellen
	CSON	Cursor auf dem LCD-Modul anzeigen
	CSOFF	Cursor auf dem LCD-Modul verbergen
	CONT	Buchstaben-Kontrast auf dem LCD-Modul einstellen
	DEC	Dezimale Numerale auf dem LCD ausgeben
	HEX	Hexadezimale Numerale auf dem LCD ausgeben
(3)	BIN	Binäre Numerale auf dem LCD ausgeben

Auf den Operand bezogene Operationen

	AND	Verwendung des logischen Ausdrucks „und“
	OR	Verwendung des logischen Ausdrucks „oder“
	MOD	Kalkulationsmodul für arithmetische Operationen
	XOR	Verwendung des logischen Ausdrucks „XOR“
(3)	NOT	Alle Bits zurücksetzen

Befehle zur Motorsteuerung

	ZERO	Einstellen des 0 Punktes (Standartwinkel) eines Stellmotors
	MOTOR	Einschalten des Ausgangsports vom Stellmotor
	MOTOROFF	Ausschalten des Ausgangsports vom Stellmotor
	MOVE	Steuerung mehrerer Motoren zum selber Zeitpunkt
	SPEED	Einstellen der Geschwindigkeit des Stellmotors
(2)	ACCEL	Einstellen der Beschleunigung des Stellmotors
	DIR	Einstellen der Drehrichtung des Stellmotors
	PTP	simultane Operationssteuerung Ein/Aus
	SERVO	Steuerung des Stellmotors
	PWM	Einstellen der Pulslänge für einen Gleichstrommotor
(2)	FASTSERVO	Servomotor mit maximaler Geschwindigkeit betreiben
(3)	HIGHSPEED	„Fast-Mode“ des Stellmotors Ein/Aus
(3)	MOVEPOS	Motor-Gruppe laut Deklaration von POS bewegen
(3)	POS	Einstellen der spezifischen Position des Roboters
(3)	FPWM	Ändern der Pulslänge und Frequenz
(3)	MOVE24	Alle 24 Stellmotoren zur gleichen Zeit bewegen
(3)	INIT	Einstellen der Anfangsbewegungshaltung
(3)	MOTORIN	Auslesen der aktuellen Positionswerte des Stellmotors
(3)	AIMOTOR	Konfiguration für das Benutzen des AI-Motors
(3)	AIMOTORIFF	Abbrechen der Nutzung des AI-Motors
(3)	AIMOTORIN	Auslesen der aktuellen Positionswerte des AI-Motors
(3)	SETON	Konfiguration zur Nutzung der Konfigurations-Funktion
(3)	SETOFF	Abbruch der Konfiguration zur Nutzung der Konfigurations-Funktion
(3)	ALLON	Konfigurations-Funktion für alle Stellmotoren
(3)	ALLOFF	Abbruch der Konfigurations-Funktion für alle Stellmotoren
(3)	GETMOTORSET	Auslesen der aktuellen Positionswerte des Stellmotors und beibehalten der aktuellen Position

Parameter, welche die Motorgruppe zuordnen

(3)	G6A	Servomotoren #0-#5 zu Gruppe A zuordnen
(3)	G6B	Servomotoren #6-#11 zu Gruppe B zuordnen
(3)	G6C	Servomotoren #12-#17 zu Gruppe C zuordnen
(3)	G6D	Servomotoren #18-#23 zu Gruppe D zuordnen
(3)	G6E	Servomotoren #24-#29 zu Gruppe E zuordnen
(3)	G8A	Servomotoren #0-#7 zu Gruppe A zuordnen
(3)	G8B	Servomotoren #8-#15 zu Gruppe B zuordnen
(3)	G8C	Servomotoren #16-#23 zu Gruppe C zuordnen
(3)	G8D	Servomotoren #24-#31 zu Gruppe D zuordnen
(3)	G12	Servomotoren #0-#11 zuordnen
(3)	G16	Servomotoren #0-#15 zuordnen
(3)	G24	Servomotoren #0-#23 zuordnen
(3)	G32	Servomotoren #0-#31 zuordnen

Befehle zur Klangkontrolle

(2)	BEEP	Warnungsgeräusch mit PIEZO erzeugen

(2)	SOUND	Frequentierten Sound mit PIEZO erzeugen
(2)	PLAY	Ein Lied mit PIEZO abspielen
(3)	MUSIK	Musik mit PIEZO abspielen
(3)	TEMPO	Rhythmus des Sounds einstellen

Befehle zur externen Kommunikation

(2)	RX	RS-232-Signal durch RX-Port empfangen
(2)	TX	RS-232-Signal durch TX-Port senden
(2)	MINIIN	Minibus-Signal durch den Mini-Kommunikationsport empfangen
(2)	MINIOUT	Minibus-Signal durch den Mini-Kommunikationsport übertragen
(3)	ERX	RS-232-Signal durch RX-Port empfangen
(3)	ETX	RS-232-Signal durch TX-Port senden

Befehle zur analogen Signalverarbeitung

(3)	AD	Analoges Signal vom AD-Port empfangen
(3)	REMOCON	Schlüsselwert vom Infrarot-Controller empfangen
(3)	SONAR	Distanz vom Ultraschallwellen-Port empfangen
(3)	RCIN	Eingabesignal vom RC-Remote-Controller empfangen
(3)	GYRODIR	Konfiguration der Richtung des Gyroskops
(3)	GYROSET	Gyroskop einem Servomotor zuordnen
(3)	GYROSENSE	Konfiguration der Empfindlichkeit des Gyroskops

Verarbeitungsbefehle

	ON…GOTO	Überspringen bei bestimmtem Wert der Variable

Sonstige Befehle

	RND	Zufallszahl generieren
	REMARK	Erzeugen eines Eintrages in Textform

Absichtsbefehle

	$DEVICE	Konfiguration des Controllers zur Nutzung von einem derzeitig laufendem Programm
(3)	LIMIT	Beschränken des Bewegungsbereichs eines Stellmotors

Kapitel 2

Die allgemeine Grammatik

von roboBASIC

Weil die Grammatik von roboBASIC auf allgemeinem BASIC basiert, ist sind die meisten Strukturen von roboBASIC ähnlich wie in der Grundsprache. In diesem Kapitel wird die allgemeine Grammatik von roboBASIC erklärt.

Zeichensatz

Der Zeichensatz von roboBASIC ist aus englischen Buchstaben (A-Z, a-z), Nummern (0-9) und speziellen Symbolen zusammengesetzt. Die in der folgenden Tabelle aufgeführten Symbole haben eine spezielle Bedeutung in roboBASIC.

Symbol	Beschreibung
+	Additionssymbol
-	Subtraktionssymbol
*	Multiplikationssymbol
/	Divisionssymbol
%	Restesymbol
.	Bitkennzeichnungssymbil
&	Ziffernsymbol
??	Textsymbol
??	Zeichenkettensymbol
:	Labelsymbol
=	Gleichheitszeichen- und Substitutionssymbol
<	Ungleichheitssymbol
>	Ungleichheitssymbol
<<	Bitveränderungssymbol
>>	Bitveränderungssymbol

Formeln und Operatoren

Formeln können aus Werten zusammengesetzt werden, welche von integrierten Invariablen, Variablen oder Konstruktionen aus allen anderen Nutzungsoperatoren berechnet werden.
Ein Operator führt Rechenoperationen oder logische Vorgänge für einen gegebenen Wert aus.
In roboBASIC können Operatoren wie in der nachstehenden Tabelle klassifiziert werden.

Klassifikation	Funktion
Arithmetischer Operator	Durchführung von Berechnungen
Relationaler Operator	Vergleich numerischer Werte
Logischer Operator	Vergleich von komplexen Bedingungen oder Ausführung von Bit-Operationen
Bitoperator	Bitveränderungen oder Ausführung von Bit-Operationen

Arithmetische Operatoren

Ein arithmetischer Operator ist ein Symbol, das eine Verarbeitung ausführt. Wie in der allgemeinen BASIC-Sprache können Addition (+), Subtraktion (-), Multiplikation (*), Division (/) und Restebildung (% oder MOD) in roboBASIC genutzt werden. Allerdings gibt es einige verschiedene Punkte zwischen roboBASIC und allgemeinem BASIC.

Erstens gibt es keine Vorränge in den Operatoren

In roboBASIC können runde Klammern nicht benutzt werden.

Beispiel: A = 1, B = 2, C = 3

Allgemeines Basic
A + B * C = 1 + 2 * 3 = 1 + 6 = 7 (in BASIC würden runde Klammern gebraucht, wenn die Addition Vorrang vor der Multiplikation haben soll)

roboBASIC
A + B * C = 1 + 2 * 3 = 3 * 3 = 9

Zweitens verursachen komplizierte mathematische Verarbeitungen unerwartete Fehler.

In diesem Fall muss die mathematische Verarbeitung in 2 oder 3 Verarbeitungen eingeteilt werden.

Beispiel:

D = A * B + C (Akzeptiert)
F = A * B / C * D + E (Vermeiden komplizierter arithmetischer Verarbeitungen)

Drittens unterstützt roboBASIC nur Byte-Typen oder ganzzahlige Datentypen, so dass ein Dezimalzeichen im Ergebnis ignoriert wird.

Modulberechnungen nutzen das „%“ oder „MOD“ Symbol und die Ausgabe erfolgt modular.

Relationale Operatoren

Ein relationaler Operator wird verwendet, um zwei Werte zu vergleichen. Ausgabe ist "TRUE" oder "FALSE". Diese Ausgabe wird für das Kontrollieren des Ablaufs des Programms in Form einer IF-Abfrage verwendet.

Operator	Beziehung	Ausdruck
=	Gleich zu	X = Y
<>	Nicht gleich	X <> Y
<	Weniger als	X < Y
>	Mehr als	X > Y
<=	Gleich oder weniger als	X <= Y
>=	Gleich oder mehr als	X >= Y

Wenn ein Rechenoperator und ein logischer Operator gemeinsam in einer Formel sind, wird der Rechenoperator vor dem logischen Operator ausgeführt.

Logische Operatoren

Ein Logikoperator wird für das Vergleichen von gemeinsamen Bedingungen verwendet. Das Ergebnis des Vergleichs wird mit „TRUE“ oder „FALSE“ beschrieben. Diese Ausgabe wird für das Kontrollieren des Ablaufs des Programms in Form einer IF-Abfrage verwendet.

Operator	Bedeutung
AND	Und
OR	Disjunktion
XOR	Exklusive Disjunktion

Jeder Operator hat eine Ausgabe wie Tabelle unten. In der Tabelle bedeutet „T“ TRUE und „F“ FALSE.

Wert von X, Y		Ausgabe		
X	**Y**	**X AND Y**	**X OR Y**	**X XOR Y**
T	T	T	T	F
T	F	F	T	T
F	T	F	T	T
F	F	F	F	F

Bit-Operatoren

Ein Bit-Operator führt Berechnungen für jede Variable die vom Robotersteuerprogramm verwendet wird aus. Die Bit-Steuerung durch die Eingangs- und Ausgabe-Ports wird dadurch vereinfacht.
Es gibt die Bit-Summe (OR), das Bit-Produkt (AND) und die exklusive Bit-Summe zur Kalkulation des gesamten Bits. In roboBASIC werden die Kalkulationszeichen links (<<), rechts (>>) und „.“ genutzt, um sich zu an einen bestimmen Punkt im Bit zu begeben.

Wenn der Wert von A 33 (Binärnummer 00100001) ist und der Wert von B 15 (Binärnummer 00001111) ist, treten die folgenden Ergebnisse auf, wenn sie die erwähnten Operatoren verwenden.

Operator	Ausgabe
A AND B	1 (00000001)
A OR B	47 (00101111)
A XOR B	46 (00101110)
A << 1	66 (01000010)
A >> 1	16 (00010000)
A.0	1 (0rd Bit von A)

Wenn mehrere Operatoren bei demselben Befehl verwendet werden, wird Betrieb in der folgenden Ordnung ausgeführt:

1) arithmetischer Operator/Bitoperator
2) relationaler Operator
3) logischer Operator

Abbildungen, Variable/Konstante und andere grammatische Erklärungen

Weil roboBASIC zur Steuerung von Hardware gestaltet ist, unterstützt roboBASIC keine Variablen oder Konstanten, die mit Zeichenfolgen verbunden sind, wie sie im Allgemeinen in BASIC verwendet werden.

Typgestaltungen

Es gibt Byte-Typabbildungen und ganzzahlige Datentypabbildungen. Der Bereich entsprechend der Art, welcher durch die Abbildung genutzt wird ist unten erkennbar.

Typ	Größe	Bereich
BYTE	1 Bytes (8bit)	0-255
INTEGER	2 Bytes (16bit)	0-65535

roboBASIC unterstützt keine negativen Zahlen. Wenn ein „+“ oder „-„ vor einer Nummer hinzugefügt wird liefert das Ergebnis der Operation einen Fehler.
Deklarationen müssen in einem geeigneten Nummerntyp erfolgen.

Antilogarithmus

Weil roboBASIC zur Hardwarekontrolle entwickelt wurde ist die Nutzung von Hexadezimalzahlen oder anderen Ausdrücken sinnvoller als die Nutzung von dezimalen Nummerntypausdrücken. In roboBASIC können Binärzahl (Bin), Oktonärzahl (Okt), Dezimalzahl (Dez), Hexadezimalnummer (Hexadezimalzahl) genutzt werden.

Antilogarithmus	Deklaration	Verwendbare Abbildung	Beispiel
Binärzahl	&B	0, 1	&B111101
Oktonärzahl	&O	0, … , 7	&O75
Dezimalzahl	N/A	0, … , 9	61
Hexadezimalzahl	&H	0, … , 9, A, … F	&H3D

Konstante und Variable

Eine Konstante ändert sich während der Programmausführung nicht. roboBASIC kann eine Konstante in Form einer Zahl vom Typ Byte oder vom Typ ganzzahlig definieren. Der Typ der Konstante wird automatisch entsprechend dem Bereich der Nummer definiert. Ist eine Konstante einmal definiert kann sie nicht mehr geändert werden. Eine Konstante zu definieren, hat keine Wirkung auf die Größe des Programms. Programmmodifikationen können angenehmer sein, wenn eine Zahl, die häufig verwendet wird, als Konstante definiert wird.

Beispiel:

```
CONST OFF = 0
CONST motor_1 = 3
CONST motor_1 speed = 200
```

Eine Variable ist der Name einer Speicherstelle in Daten, die innerhalb des Programms verwendet werden. Im robonova Steuerprogramm ist die Anzahl von Variablen beschränkt, so dass die Variablendeklaration dafür entworfen werden muss, die Größe der Variablen entsprechend dem Objekt zu reduzieren.

```
DIM motor_1_delay AS INTEGER
DIM sensor_left AS BYTE
```

Bei der Deklaration einer Konstanten oder einer Variable beachten sie die folgenden Regeln:

Erstens muss Englisch oder Koreanisch beim ersten Buchstaben benutzt werden. Bei Koreanisch (Chinesisch) oder Englisch können Zahlen und "_" für den Variablen- oder Konstantenamen verwendet werden.

Zweitens darf der Variablen- oder Konstantenname nicht länger als 64 Zeichen sein.

Drittens dürfen Konstanten- und Variablenname nicht übereinstimmen. Weiterhin gibt es keine Unterscheidung zwischen Klein- und Großbuchstaben.

Viertens können bei der Zuweisung von Werten größer als 65535 (Grenze des ganzzahligen Bereichs) zu Konstanten Probleme bzw. Fehler auftreten.

Bits zeigen

In roboBASIC können Variablen als Bit-Einheiten gehandhabt werden. Um Variablen als eine Bit-Einheit zu nutzen wird der zeigende Bit-Operator „.“ Genutzt. Bei der Benutzung des Bit-Operators sind Bits 0~6 (Byte-Variable) und Bits 0~16 (ganzzahlige Variable) nutzbar. Allerdings sind nur Zahlen oder Konstanten mit diesem Operator nutzbar.

Beispiel:

```
DIM A AS INTEGER
	CONST BIT_2 = 2

	A.1 = 1
	A.BIT_2 = 0
	A.3 = IN(1)		Wert von Port #1 auslesen und in das dritte Bit der ganzzahligen
Variable A schreiben

	OUT 2, A.1		Den Wert des ersten Bits der ganzzahligen Variable A mit Port
#2 ausgeben
```

Erklärungsanweisungen
Codeerklärungen sollten innerhalb des Programms eingefügt werden um damit effizient zu arbeiten und es zu bearbeiten. Um eine Erklärung einzufügen wird das Symbol „'“ oder der Befehl REMARK genutzt. Der Ort eines Kommentars innerhalb eines Programms hat keinen Einfluss auf die Ausführung des Programms.

Substitutionsanweisung (=)
Die Substitutionsanweisung wird dafür genutzt Variablen einen bestimmten Wert zuzuweisen. Dazu wird das Symbol „=“ genutzt. Der Name der Variable steht dabei immer auf der linken Seite des Substitutionssymbols und auf der rechten Seite findet man Variablen, Zeichenketten oder Funktionen.

Beispiel:

A = B	*Substitution von Variablen*
A.1 = 1	*Bit-Substitution*
A = ADIN(0)	*Substitution einer Funktion*
A = 3 * 2 – 1	*Substitution von Zahlen*
A = C + B – A	*Substitution von Buchstaben*
A = „1“	*Substitution von ASCII-Code*

Sprungmarken
Eine Sprungmarke (Label) kann dazu genutzt werden um zu einem bestimmten Punkt in einem Programm zu springen. Es können Buchstaben und Zahlen zur Bezeichnung einer Sprungmarke genutzt werden. Aber es gibt einige Regeln zur Setzung von Sprungmarken.
Erstens, darf die Länge der Bezeichnung eines Labels keine 64 Zeichen überschreiten und das erste Zeichen muss in Englisch oder Koreanisch sein.
Zweitens muss das Label-Symbol „:“ unmittelbar nach der Bezeichnung folgen.
Drittens können Numerale innerhalb des 0 ~ 65535 Bereichs für die Bezeichnung von Sprungmarken genutzt werden. Das Label-Symbol wird dann nicht mehr benötigt.
Viertens können Sprungmarkennamen nicht doppelt genutzt werden und es gibt keine Unterscheidung zwischen Klein- und Großbuchstaben.
Meistens wird ein Label zur Flusskontrolle innerhalb eines Programms zusammen mit den Befehlen GOTO oder GOSUB genutzt.

Beispiel:

```
DIM A AS INTEGER
START:
        A = IN(0)
        IF A = 0 THEN
                GOTO START
        ELSE
                GOSUB 10
        END
        GOTO START
10 OUT 1, 0
        DELAY 100
        OUT 1, 1
        RETURN
```

Kapitel 3

Erklärung der Deklaration

von Befehlen für roboBASIC

Dieser Befehl wird zur Deklaration einer Variable oder Konstante genutzt.

DIM … AS Deklaration … als

Deklaration einer Variablen

Befehlssatz

1) Im Falle der Deklaration einer einzelnen Variable
   - DIM [Variablenname] AS [Variablentyp]

2) Im Fall der Deklaration von mehreren Variablen
   - DIM [Variablenname] AS [Variablentyp], [Variablentyp] AS [Variablentyp] ...

Erklärung des Befehls

Eine Variable, welche in roboBASIC genutzt wird, muss mit dem DIM-Befehl deklariert werden. Der DIM-Befehl muss in Verbindung mit AS genutzt werden um den Variablentyp festzulegen. Beim Variablennamen gibt es keine Unterscheidung zwischen Groß- und Kleinbuchstaben. Variablennamen dürfen nicht doppelt auftreten.
Eine Variable wird verwendet um den Wert eines Sensors oder umgewandelte analoge Werte zu verarbeiten. Die Verwendung passender Variablen macht damit die Programmgestaltung effizienter. Die Anzahl der nutzbaren Variablen hängt von der verwendeten Roboter-Kontrolleinheit ab.
Die MR-C2000 Serie nutzt Variablen mit weniger als 30 Byte Größe. Die MR-C3000 Serie nutzt Variablen mit weniger als 256 Byte Größe. Byte-Typ-Variablen haben immer die Größe von 1 Byte und Integer Variablen belegen eine Größe von 2 Bytes. Die Deklaration von Variablen sollte daher immer korrekt erfolgen, um die maximale Anzahl von Variablen nicht sinnlos zu beschränken.

Beispiel des Befehls

```
DIM I AS INTEGER        Deklaration von I als Integer
DIM J AS BYTE           Deklaration von J als Byte
```

CONST

Konstante deklarieren

Befehlsstruktur

CONST [Konstantenname] = Zahl

Erklärung des Befehls

Die Vergabe eines Konstantennames für eine Zahl vereinfacht den Programmierprozess erheblich.

Einige Vorteile der Benutzung von Konstanten anstatt von Zahlen oder Variablen sind:

1.) Wenn eine Konstante einmal definiert wurde kann sie im ganzen Programm genutzt werden
2.) Eine Konstante kann durch Fehler nicht geändert werden
3.) Die Modifikation gestaltet sich sehr einfach
4.) Eine Konstante verbraucht nur sehr wenig Speicher

Beispiel des Befehls

CONST OFF = 0	Deklaration der Konstante OFF als 0
CONST A = &HB1001	Deklaration der Konstante A als Dezimalzahl 9

Kapitel 4

Ablaufsteuerung

Befehlserklärung

Diese Befehle werden verwendet, um Programmablauf zu kontrollieren.

IF … THEN …

Befehlsstruktur

1.) Einfache Bedingung:
    - IF [Bedingung] THEN [Anweisung, wenn die Bedingung wahr ist] ENDIF

2.) Mehrere Bedingungen:
    - IF [Bedingung 1] THEN
      [Anweisung, wenn Bedingung 1 wahr ist]
      ELSEIF [Bedingung 2] THEN
      [Anweisung, wenn Bedingung 2 wahr ist]
      ELSE
      [Anweisung, wenn Bedingung 1 und Bedingung 2 falsch sind]
      ENDIF

Erklärung des Befehls

Bei der Ausführung eines IF … THEN Befehls wird die IF Bedingung überprüft. Wenn die Bedingung TRUE ist, dann werden die Befehle ausgeführt. Wenn die Bedingung jedoch FALSE ist wird jede Folgebedingung überprüft und ausgeführt. Ansonsten wird der ELSE Teil der Bedingung ausgeführt. Hier kann wiederum ein ELSEIF eingeschlossen werden. Dies ist allerdings nicht zwingend notwendig.
In roboBASIC ist der IF … THEN Befehl unbedingt notwendig um in Verbindung mit externen Eingaben einen externen Wert in eine Variable zu schreiben. Die bedingte Anweisung beurteilt hierbei den Wert der Variablen und ermöglicht eine Bewegung des Roboters entsprechend den Werten.

Beispiel des Befehls

1.) Die Ausführung der Bedingung und der Anweisungen ist sehr einfach. Beide können auf derselben Zeile angegeben werden.

```
IF A > 0 THEN B = 5
IF A < 5 THEN B = 0 ELSE B = 1
```

2.) Die Bedingung eines IF Befehls kann bei der Nutzung von Relationszeichen 2 verschiedene Bedingungen beachten

```
IF A > 0 AND A < 5 THEN B = 3
IF A = 7 OR A = 9 THEN B = 1
```

3.) Beispiel einer verschachtelten IF-Anweisung

```
IF A = 1 THEN
        B = 2
        C = 3
ELSEIF A = 3 AND A = 5 THEN
        B = 1
        C = 2
ELSEIF A = 8 THEN
        B = 6
        C = 0
ELSE
        B = 0
        C = 0
ENDIF
```

FOR … NEXT

Wiederholt sich für eine festgelegte Anzahl an Durchgängen

Befehlsstruktur

```
FOR [Schleifenvariable] = [Start] TO [Ende]
        [Schleifenanweisungen]
NEXT [Schleifenvariable]
```

Erklärung des Befehls

Die [Schleifenvariable] enthält die Anzahl der Schleifendurchläufe. [Start] ist hierbei der Startwert der Schleife und [Ende] ist der letzte Wert der Schleifenvariable. Es können Zahlen, Konstanten oder Variablen für die [Start] und [Ende] Werte genutzt werden.
Bei roboBASIC muss der [Ende]-Wert größer als der [Start]-Wert sein. RoboBASIC vergrößert immer die Zählvariable der Schleife. Weiterhin gibt es einige Regeln zur Benutzung von FOR … NEXT Schleifen:

1. Eine FOR … NEXT Anweisung kann in einer anderen genutzt werden.

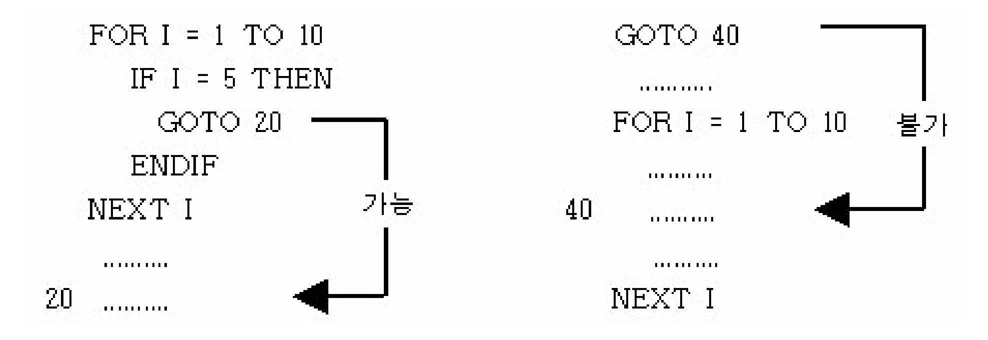
```
FOR I = 1 TO 10
        FOR J = 1 TO 5
                ..........
        NEXT J
NEXT I
```

2. Bei der Nutzung mehrer FOR … NEXT Anweisungen muss die Anordnung von NEXT [Schleifenvariable] nicht geändert werden.

```
FOR I = 1 TO 10
        FOR J = 1 TO 5
                ..........
        NEXT I
NEXT J
```

   Allerdings sind in diesem Beispiel die Schleifenvariablen vertausch worden. Ein Fehler sollte beim programmieren nicht auftreten, allerdings kann es bei der Ausführung auf der robonova-Kontrolleinheit zu unerwünschten Ergebnissen kommen.

3. Es ist kein Problem die Ausführung eines Befehls innerhalb einer FOR … NEXT Anweisung zu beenden, aber man sollte beachten eine FOR … NEXT Anfrage nicht von außen zu beginnen!



4. Die Werte, welche für die [Schleifenvariable], [Start] und [Ende] genutzt werden dürfen während der Ausführung der Schleife nicht geändert werden.

Beispiel des Befehls
Eine LED, welche an den Port #0 der Kontrolleinheit angeschlossen ist soll fünfmal blinken.

```
DIM A AS BYTE       Variable zur Nutzung in der Schleife deklarieren

FOR A = 1 TO 5      Die Anzahl der Wiederholungen ist 5
OUT 0, 0            LED, welche an Port #0 angeschlossen ist einschalten
    DELAY 100       Verzögerung von 100
OUT 0, 1            LED, welche an Port #0 angeschlossen ist ausschalten
    DELAY 100       Verzögerung von 100
NEXT A
```

GOTO
Zu einem bestimmten Punkt springen

Befehlsstruktur
GOTO [Sprungmarke]

Erklärung des Befehls
Der GOTO-Befehl verändert den Ablauf des Programms durch springen zu einer bestimmten Zeile im Programm-Code. Die Benutzung des GOTO-Befehls verkompliziert den Programmaufbau. Daher sollte man diesen Befehl nicht zu oft verwenden.

Beispiel des Befehls
```
        DIM I AS INTEGER
        DIM J AS BYTE

        I = 7
        IF I = 6 THEN GOTO L1
            ................
L1: J = 1
        OUT I, J
```

GOSUB … RETURN
Eine Subroutine aufrufen und wieder zurückkehren

Befehlsstruktur
```
    GOSUB [Sprungmarke]
        ……...............
[Sprungmarke]: ……...............
    RETURN
```

Erklärung des Befehls
Die GOSUB-Routine ruft einen häufig genutzten Programmteil auf und kehrt dann zum eigentlichen Programm zurück! Auf diesem Weg kann man das Programm kleiner und effizienter gestalten.

Man kann innerhalb einer Unterroutine eine weitere aufrufen. Die MR-C2000-Serie unterstützt dabei maximal 4 Routinen und die MR-C3000-Serie 5 Unterroutinen. Eine größere Anzahl verursacht Fehler.

Beispiel des Befehls

```
DIM LED_PORT AS INTEGER
LED_PORT = 1
START: .............
.............
GOSUB LED_TOGGLE
...............
GOTO START
END

LED_TOGGLE:
    TOGGLE LED_PORT
    RETURN
```

END

Beendet die Ausführung des Programms

Befehlsstruktur

```
END
```

Erklärung des Befehls

2 Sekunden nach dem Einschalten der robonova-Kontrolleinheit wird das Programm aus dem EEPROM ausgeführt. Wenn der END-Befehl nicht am Ende einer Unterroutine oder eines Befehls innerhalb eines Programms genutzt wird führt sich das Programm permanent aus. Deshalb sollte man unbedingt einen END-Befehl am Ende jedes Befehls bzw. jeder Unterroutine nutzen.

Beispiel des Befehls

1. Die Programmausführung beenden

```
DIM A AS BYTE
START: A = IN(0)
    IF A = 1 THEN END
    ...........
    GOTO START
```

2. Erstellung einer strukturmäßigen Unterroutine

```
DIM A AS BYTE

A = BYTEIN(0)
IF A = 1 THEN
    GOSUB L1
ELSEIF A = 3 THEN
    GOSUB L2
ELSEIF A = 4 THEN
    GOSUB L3
ELSE
    GOSUB L4
ENDIF
...........
END
```

```
L1: .............
        RETURN
L2: ............
        RETURN
L3: ...........
        RETURN
L4: ..........
        RETURN
```

STOP/RUN

Startet oder beendet die Programmausführung

Befehlsstruktur

STOP / RUN

Erklärung des Befehls

Dieser Befehl startet oder stoppt die Ausführung des Programmes. Nach der Benutzung des STOP-Befehls kann das Programm mit RUN erneut initialisiert bzw. gestartet werden.

WAIT

Warten bis das Programm vollständig ausgeführt wurde.

Befehlsstruktur

WAIT

Erklärung des Befehls

Das Befehlssystem in den Roboter-Kontrolleinheiten hat eine sehr gute Realzeitkontrolle.

Wenn ein Programm ausgeführt wird wird das nächste Programm direkt ausgeführt ohne das laufende Programm zu stoppen. Um das neue Programm alleine nach dem anderen Programm auszuführen wird der WAIT-Befehl genutzt.

Beispiel des Befehls

Beispiel 1:
Ausgabe an den Ports 7 und 8 nach der Bewegung von 6 Motoren.

```
MOVE 120, 100, 140, 90, 70, 150
WAIT
OUT 7, 1
OUT 8, 1
```

Beispiel 2:
Ausgabe an Port 8 nach der Bewegung, aber an Port 7 während der Bewegung

```
MOVE 120, 100, 140, 90, 70, 150
OUT 7, 1
WAIT
OUT 8, 1
```

DELAY
Verzögern der Programmausführung um eine festgelegt Zeit

Befehlsstruktur

DELAY [Verzögerungsdauer]

Erklärung des Befehls

Dieser Befehl verzögert die Ausführung eines Programmes um eine festgelegte Zeit. Die Verzögerungszeit kann bei Kontrolleinheiten der MR-C2000-Serie in 10ms und bei der MR-C3000 in 1ms angegeben werden.

Für die [Verzögerungsdauer] können Ziffern, Konstanten oder Variablen genutzt werden.

Beispiel des Befehls

MR-C2000-Serie:

DELAY 10 Verzögerung von 100ms (10ms*10 = 100ms = 0,1 sec.)

MR-C3000-Serie:

DELAY 500 Verzögerung von 500ms (1ms * 500 = 500ms = 0,5sec.)

BREAK
Programmausführung unterbrechen und in Debug-Modus wechseln
Nur MR-C2000-Serie.

Befehlsstruktur

BREAK

Erklärung des Befehls

Dieser Befehl führt zu einer Unterbrechung der Programmausführung und wechselt in den Debug-Modus. Wenn das Programm angehalten wird werden alle Speicherwerte der Miniroboter-Kontrolleinheit an den PC gesendet. Stellen Sie sicher, dass die Kontrolleinheit mit dem PC verbunden ist. Ansonsten wird das Programm angehalten und verbleibt in diesem Zustand. Nähere Informationen finden sie in den „Erklärungen zu roboBASIC Programmen". Der BREAK-Befehl funktioniert nicht mit Kontrolleinheiten der MR-C3000-Serie. Bei der Benutzung dieser kann der Programmablauf systematisch im Debug-Modus verfolgt werden.

Beispiel des Befehls

….
BREAK Programmausführung unterbrechen

ACTION [Nummer]
Führt vorgefertigt Bewegungen entsprechend dem Wert [Nummer] aus.

Befehlsstruktur

ACTION [Nummer]

Erklärung des Befehls
Führt vorgefertigt Bewegungen entsprechend der [Nummer] aus. Maximal sind 32 Bewegungen verfügbar.

Beispiel des Befehls

ACTION 3	Bewegung Nummer 3 ausführen
ACTION 5	Bewegung Nummer 5 ausführen
ACTION 23	Bewegung Nummer 23 ausführen

Bemerkung:
Dieser Befehl ist nur für den MR-C3024-Controller oder den Robonova I-Robot.

goto AUTO

Zum Vorlagenprogramm bewegen

Befehlsstruktur

```
goto AUTO
```

Erklärung des Befehls
Befehl um vorgefertigte Vorlagenprogramme auszuführen.

Beispiel des Befehls

goto AUTO	zum Vorlagenprogramm bewegen

Bemerkung:
Dieser Befehl ist nur für den MR-C3024-Controller oder den Robonova I-Robot.

Kapitel 5

Erläuterung

der digitalen Signaleingabe und – ausgabe

in roboBASIC

Die MR-C2000 Serie besitzt 12 digitale I/O-Ports. Bei der MR-C3000 Serie sind 40 digitale I/O-Ports vorhanden. Diese Ports können unterschiedliche Befehle ausführen. Für weitere Informationen nutzen sie bitte die „Steuereinheiterklärungen“.

IN()

Liest digitale Werte von einem Port

Befehlsstruktur

IN([Portnummer])

Erklärung des Befehls

Ein Signalwert, welcher durch einen Port eingelesen wird, wird als Variable gespeichert. Die Werte werden mit 0 oder 1 beschrieben. Byte- oder ganzzahlige Typen können ebenfalls verwendet werden. Dabei ist nur der letzte Wert des Bits verfügbar. Der effizienteste Weg ist die Nutzung von Byte-Variablen.

Beispiel des Befehls

```
DIM A AS BYTE
A = IN(0)       Signal vom Port #0 einlesen und in die Variable A schreiben
```

OUT()

Sendet digitale Signale zu einem Port

Befehlsstruktur

OUT [Portnummer], [Ausgabewert]

Erklärung des Befehls

Sendet ein Signal vom Kontroller durch einen Port. Beim Senden eines Wertes von 0 wird ein 0V-Signal ausgegeben. Beim Senden eines Wertes von 1 wird ein 6V-Signal ausgegeben. Zahlen (0 oder 1), Konstanten und Variablen können für den [Ausgabewert] genutzt werden. Weiterhin können Bits für den [Ausgabewert] genutzt werden, weil sie nur 0 oder 1 enthalten.

Beispiel des Befehls

Dieses Beispiel wurde zum Test für Eingabe- und Ausgabeports geschrieben. Ein Taster ist an Port #0 angeschlossen und eine LED an Port #3.

```
DIM A AS INTEGER

A = 0               Variable A initialisieren
START: A = IN(0)    Die Bedingung des Tasters auslesen A.0 = IN(0) ist verfügbar

IF A = 1 THEN       Wenn der Taster nicht gedrückt ist
    OUT 3, 0        LED ausschalten
ELSE                sonst
    OUT 3, 1        LED einschalten
ENDIF
GOTO START          Taster erneut überprüfen
```

BYTEIN()

Befehlsstruktur

BYTEIN([Byte-Portnummer])

Erklärung des Befehls
Der Roboterkontrollerport kann Eingaben/Ausgaben als eine Einheit behandeln. In einigen Fällen muss das Signal Ein-/Ausgabe eine Einheit darstellen (wie hier die Einheit an Port #0).

MR-C2000-Kontroller:

Byte-Port	Port
0	Ports #0~#7 (#0 ist ein Port niedriger Ordnung)
1	Ports #8~#11 (#8 ist ein Port niedriger Ordnung)

MR-C2000-Kontroller:

Byte-Port	Port
0	Ports #0~#7 (#0 ist ein Port niedriger Ordnung)
1	Ports #8~#16 (#8 ist ein Port niedriger Ordnung)
2	Ports #16~#23 (#16 ist ein Port niedriger Ordnung)
3	Ports #24~#31 (#24 ist ein Port niedriger Ordnung)
4	Ports #32~#39 (#32 ist ein Port niedriger Ordnung)

Bei der Benutzung des BYTEIN Befehls wird ein Signalwert, welcher durch den Byte-Eingangsport empfangen wird, als Variable gespeichert. Die Variable muss als Byte- oder ganzzahliger Typ deklariert sein.

Beispiel des Befehls
A = BYTEIN(0)   Alle Signale der Ports #0~#7 sind Eingaben der Variable A

BYTEOUT
Gibt ein Signal an einem Port als Byte-Einheit aus

Befehlsstruktur
BYTEOUT [Byte-Portnummer], [Ausgabewert]

Erklärung des Befehls
Ausgabesignalwerte durch den Byte-Einheit-Port, Numerale, Konstanten oder Variablen können als [Byte-Portnummer] genutzt werden. Für den [Ausgabewert] können Numerale zwischen 0~255, Konstanten oder Byte-Variablen genutzt werden.

Beispiel des Befehls
BYTEOUT 0, 255   Den Wert von 1 (5V) an die Ports #0~#7 senden
BYTEOUT 0, &h10101010
Den Wert von 1 (5V) an die Ports #1, #3, #5, #7 senden
Den Wert von 0 (0V) an die Ports #0, #2, #4, #6 senden

INKEY
Liest eine Tasteneingabe an einem Eingangsport ein

Befehlsstruktur
INKEY ([Portnummer])

Erklärung des Befehls
Wenn ein Schalter einmal gedrückt wird sind es tatsächlich hunderte oder tausende von elektrischen bzw. mechanischen Betätigungen. Dieses Phänomen wird chattering genannt. Dieses chattering kann Fehler verursachen. Deshalb sind spezielle Schutzmechanismen nötig. Im Roboter-Kontroller ist eine Schutzfunktion in die Software integriert. Um diese Software auszuführen sollte der INKEY-Befehl genutzt werden.

Beispiel des Befehls
Wert eines Schalters an Port #0 in Variable A schreiben

```
DIM A AS BYTE

A = INKEY(0)
```

STATE()
Liest den aktuellen Wert eines Ausgangsports

Befehlsstruktur
STATE ([Portnummer])

Erklärung des Befehls
Falls ein Bedingungswert an einem Ausgangsport nach dem Senden eines Signals durch den Port nötig ist benutzen sie die STATE-Funktion. Die IN-Funktion sollte dann nicht genutzt werden.

Beispiel des Befehls
Dies ist ein Beispielprogramm um die Ausgangsbedingung an Port #1 zu testen.

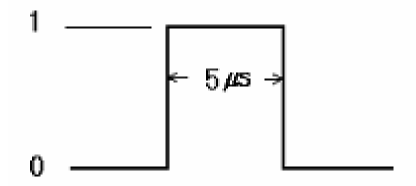
```
DIM A AS BYTE

OUT 1, 1
A = STATE(1)      A = 1
OUT 1, 0
A = STATE(1)      A =0
```

PULSE
Sendet ein pulsierendes Signal an den Ausgabe-Port

Befehlsstruktur
PULSE [Portnummer]

Erklärung des Befehls
Sendet ein pulsierendes Signal an den Ausgabe-Port. Das pulsierende Signal wird für die Bereitstellung eines Signals an ein externes Gerät genutzt.



Numerale, Konstanten oder Variablen können für die [Portnummer] genutzt werden.

Beispiel des Befehls

PULSE 3 Pulsierendes Signal zu Port #3 senden

TOGGLE

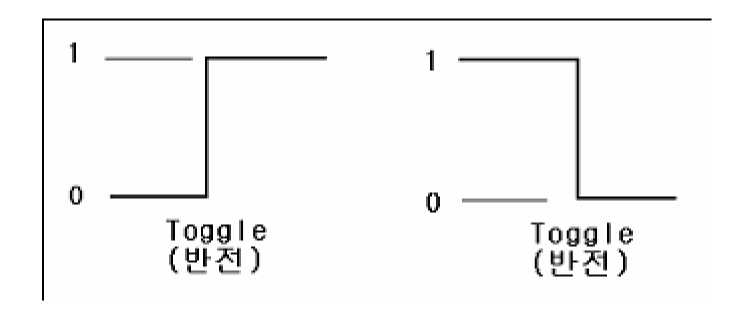
Ausgangsport-Signal umkehren

Befehlsstruktur

TOGGLE [Portnummer]

Erklärung des Befehls

Der Befehl kehrt das Signal an einem Ausgangsport um. Wenn das Signal des Ports 0 ist wird es nach der Umkehrung als 1 dargestellt.



Numerale, Konstanten oder Variablen können für die [Portnummer] genutzt werden.

Beispiel des Befehls

OUT 3, 1 Signal 1 an Port #3 senden
TOGGLE 3 Signal von Port #3 umkehren

KEYIN()

Eingabe mehrerer Tasten (analoge Tastatur)

Befehlsstruktur

KEYIN ([Analoge Portnummer], [Anzahl der Zeichen])

Erklärung des Befehls

Dieser Befehl liest die Werte von 16 Tasten durch einen AD-Konverter-Port (analoger Port) bei Steuereinheiten der MR-C3000-Serie aus.

Numerale (0~7), Konstanten und Byte-Variablen können für die [Analoge Portnummer] genutzt werden.
Der Wertebereich für eine Taste reicht von 0 bis 16. 0 bedeutet, dass die Taste nicht gedrückt ist. Numerale von 1 bis 16 bedeuten, dass die Taste gedrückt ist.

<u>Beispiel des Befehls</u>
DIM K AS BYTE

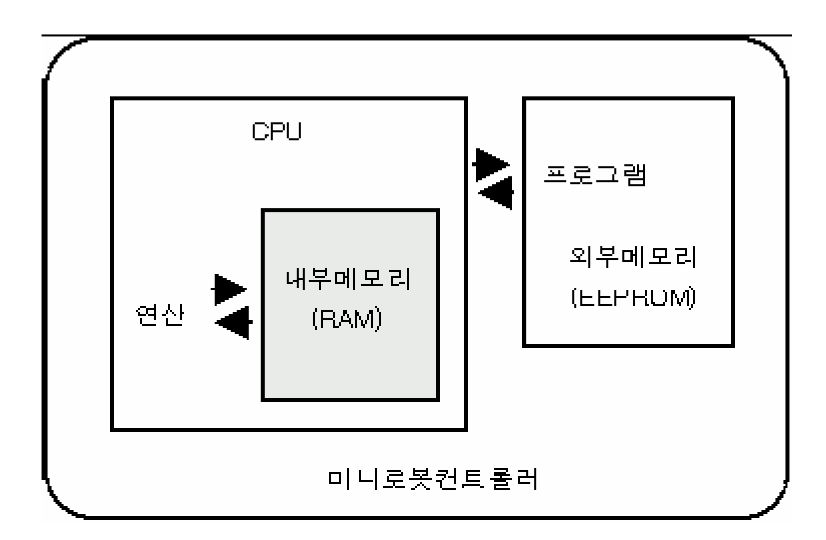
K = KEYIN(0, 16)	Einlesen der Werte von 16 Tasten, die am analogen Port #0 hängen als K

Kapitel 6

Erklärung der Befehle

zur Speichersteuerung

Die Robotersteuereinheit hat CPU und Speicher, so dass sie als Mikrocomputer bezeichnet werden kann. Speicherausführungen sind daher ein wichtiger Aufgabenbereich beim Sichern und dem Berechnen von Programmen. Der externe Speicher für die Kontrolleinheit findet sich in Form des EEPROM und wird für das Speichern von Nutzerprogrammen genutzt. Der Arbeitsspeicher befindet sich dagegen direkt in der CPU.



Der interne Speicher wird als RAM bezeichnet. Allerdings kann man ihn auch als Datensatz bezeichnen, da der spezielle Funktionen der Roboter-CPU hat. Der interne Speicher steht im Zusammenhang mit der Anzahl der verwendbaren Variablen während der Entwicklung eines Programms. Die Controller der MR-C2000-Serie haben 30Bytes variablen Speicherplatz. Die MR-C3000-Serie hat 256Bytes variablen Speicher. Die anderen Teile des Speichers sind für die interne Verwendung der Robotersteuereinheit gedacht.

Der externe Speicher spielt für die Größe der entwickelten Software eine Rolle. Die MR-C2000-Serie hat 4KBytes und die MR-C3000-Serie 12k, 32, oder 64k Bytes Speicher je nach Modell.

PEEK()

Liest den Inhalt des internen Speichers

Befehlsstruktur
PEEK([Speicherregion])

Erklärung des Befehls
Die PEEK-Funktion ruft Daten aus dem internen Speicher auf. Diese Funktion sollte nicht benutzt werden, wenn die genaue Struktur des internen Speichers nicht bekannt ist.

MR-C2000-Kontroller:
Numerale zwischen 0~255, Konstanten oder Byte-Variablen können genutzt werden.

MR-C3000-Kontroller:
Numerale zwischen 0~65535, Konstanten oder Byte-Variablen können genutzt werden.

Beispiel des Befehls
DIM A AS BYTE

A = PEEK(43)		Wert der RAM-Region 43 in Variable A schreiben

POKE

Daten in internen Speicher schreiben

Befehlsstruktur
POKE [RAM-Region], [Daten]

Erklärung des Befehls
Der POKE-Befehl wird genutzt um Daten in den internen Speicher zu schreiben.
Beim MR-C2000-Kontroller können Numerale zwischen 0~255, Konstanten oder Byte-Variablen als [RAM-Region] genutzt werden.
Beim MR-C3000-Kontroller können Numerale zwischen 0~65535, Konstanten oder Byte-Variablen als [RAM-Region] genutzt werden.

Numerale, Konstanten oder Variablen (ganzzahlige Variablen) können für [Daten] genutzt werden.

Beispiel des Befehls
POKE &h40, 100		100 in die RAM-Region 40 schreiben

ROMPEEK()
Daten vom externen EEPROM lesen

Befehlsstruktur
ROMPEEK ([ROM-Region])

Erklärung des Befehls
Die Robotersteuereinheit nutzt den EEPROM zur Speicherung von Programmen oder anderen Objekten. Die ROMPEEK oder ROMPOKE-Funktionen, welche den externen Speicher steuern können zur Datenspeicherung genutzt werden. Falls ein Speicherbereich bereits Daten enthält entsteht ein Fehler. Numerale, Konstanten oder Byte-Variablen können für die [ROM-Region] genutzt werden.

ROMPOKE
Daten in externen Speicher schreiben

Befehlsstruktur
ROMPOKE [ROM-Region], [Daten]

Erklärung des Befehls
Die Robotersteuereinheit nutzt den EEPROM zur Speicherung von Programmen oder anderen Objekten. Die ROMPEEK oder ROMPOKE-Funktionen, welche den externen Speicher steuern können zur Datenspeicherung genutzt werden. Falls ein Speicherbereich bereits Daten enthält entsteht ein Fehler. Numerale, Konstanten oder Byte-Variablen können für die [ROM-Region] genutzt werden. Numerale zwischen 0~255, Konstanten oder Byte-Variablen können für [Daten] genutzt werden.

Kapitel 7

Ansteuerung von LCD-Modulen

in roboBASIC

Das LCD-Modul, welches speziell für die Nutzung mit den Robotersteuereinheiten abgestimmt ist trägt die Bezeichnung MR-16202.
Das LCD-Modul wird bei der MR-C2000-Serie an Port #6 angeschlossen. Die MR-C3000-Serie hat einen speziellen LCD-Port.
Die Befehle zur Steuerung des LCD-Moduls und zur Anzeige von Zeichenketten werden im Folgenden erklärt.



MR-16202 LCD-Modul

LCDINIT

Initialisiert das LCD-Modul

Befehlsstruktur
LCDINIT

Erklärung des Befehls
Das LCD-Modul muss durch die Nutzung des LCDINIT-Befehls initialisiert werden, um zu verhindern, dass ungewollte Zeichen angezeigt werden. Bei der Initialisierung werden alle Zeichen auf dem LCD gelöscht und der Zeiger an den linken Displayrand an den Anfang gesetzt.

Beispiel des Befehls
LCDINIT LCD-Modul initialisieren

CLS

Zeichen im LCD-Modul löschen

Befehlsstruktur

CLS

Erklärung des Befehls

Um alle Zeichen im LCD-Modul zu löschen wird der CLS-Befehl genutzt. Wenn der CLS-Befehl ausgeführt wird werden alle Zeichen gelöscht und der Zeiger wird an den Anfang gesetzt. Allerdings gibt es Unterschiede zwischen LCDINIT und CLS. Beim CLS-Befehl werden nur die Zeichen auf dem LCD gelöscht. Beim LCDINIT-Befehl dagegen werden alle Informationen, also auch interne Variablen gelöscht.

Beispiel des Befehls

CLS Alle Anzeigen auf dem LCD-Modul löschen

LOCATE

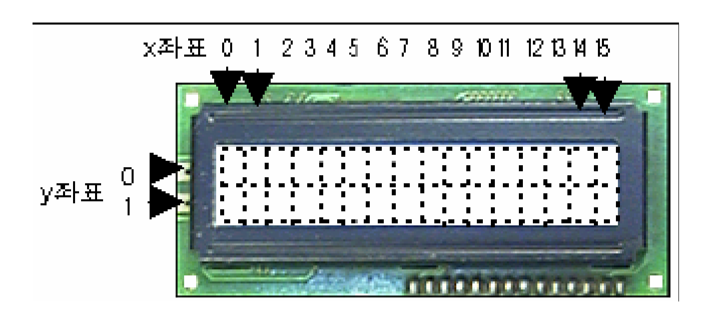
Auf die Anzeigeposition eines Zeichens im LCD-Modul zeigen

Befehlsstruktur

LOCATE [x-Koordinate], [y-Koordinate]

Erklärung des Befehls

Mit dem LOCATE-Befehl wird auf einen Punkt entsprechend den x- und y-Koordinaten gezeigt. Die Koordinaten in einem 16x2 LCD-Modul sind wie im Bild unten angeordnet. Numerale, Konstanten und Variablen können für die Koordinaten x und y genutzt werden. Allerdings muss bei x und y = 0 gestartet werden.



Beispiel des Befehls

LOCATE 0, 0 Zeiger an den Anfang des LCD-Moduls setzen
LOCATE 4, 1 Zeiger auf die Koordinaten (4, 1) des LCD-Moduls setzen

PRINT

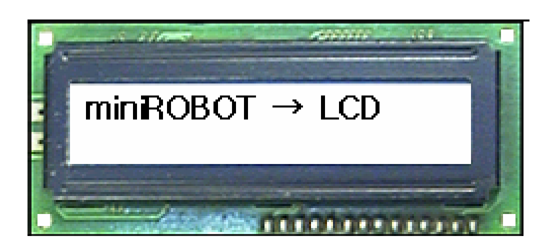
Ausgabe von Zeichen auf dem LCD-Modul

Befehlsstruktur

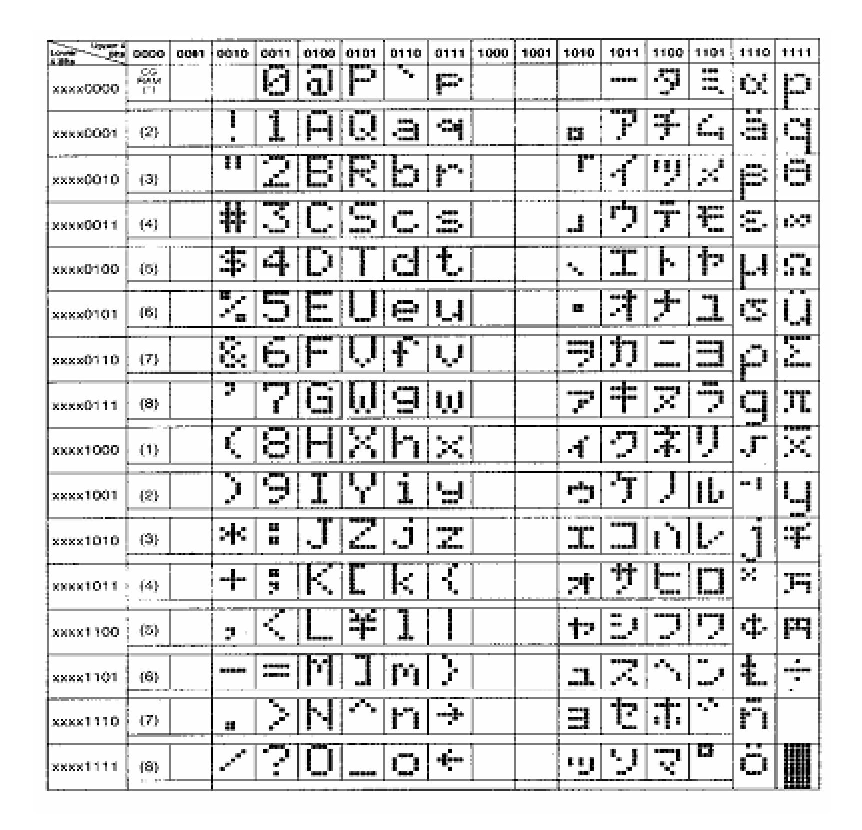
PRINT „[Zeichenkette]“, [Numeral]/ „[Zeichenkette]]“, …

Erklärung des Befehls
Um ein Zeichen an die aktuelle Stelle des Zeigers auszugeben wird der PRINT-Befehl genutzt. Die [Zeichenkette] kann durch doppelte Anführungszeichen („_“) gekennzeichnet werden. Der Bereich der Numerale liegt zwischen 1~255 (die 0 kann nicht genutzt werden). Im LCD-Modul werden alle Zeichen die auf ASCII-Code zurückzuführen sind angezeigt.

Beispiel des Befehls
CLS
PRINT „robonova, 126, „LCD“



Die folgenden Beispiele sind welche mit ASCII-Code für ein 16x2-Zeilen-LCD-Modul. Der Zeichencode hängt vom verwendeten LCD-Modul ab.



FORMAT()
DEC()
HEX()
BIN()

Spezifikation des Typs eines auf dem LCD-Modul angezeigten Numerals

Befehlsstruktur

FORMAT ([Variable], [Ausgabetyp], [Position des Punktes])

Erklärung des Befehls

Das LCD-Modul folgt einem bestimmten Format, wenn es eine Variable ausgibt. Nach einem PRINT-Befehl muss unbedingt ein FORMAT-Befehl folgen.

[Variablentyp] / [Ausgabetyp]	Standartposition des Punktes	Zahlenausdruck
Byte-Typ / dezimal	3	0 ~ 255
Byte-Typ / hexadezimal	2	00 ~ FF
Byte-Typ / binär	8	00000000 ~ 11111111
Integer-Typ / dezimal	5	0 ~ 65535
Integer-Typ / hexadezimal	4	0000 ~ FFFF
Integer-Typ / binär	16	0000000000000000 ~ 1111111111111111

Beispiel des Befehls

```
DIM A AS BYTE
DIM B AS INTEGER

LCDINIT
A = 100
B = 20000
LOCATE 0, 0
PRINT FORMAT(A, DEC, 4)
LOCATE 0, 1
PRINT FORMAT(B, HEX)
```



CSON()
CSOFF()

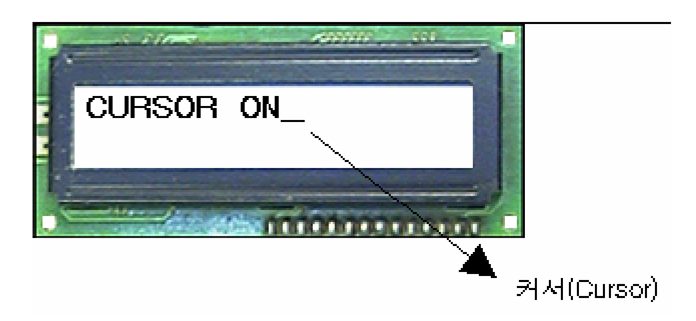
Zeiger auf dem LCD-Modul anzeigen/verstecken

Befehlsstruktur

CSON / CSOFF

Erklärung des Befehls
DIE CSON/CSOFF-Befehle zeigen bzw. verstecken den Zeiger auf dem LCD-Modul. Standardmäßig ist der Zeiger versteckt, wenn das LCD-Modul initialisiert wird.

Beispiel des Befehls
LCDINIT
CSON
PRINT "CURSOR ON"



CONT
Den Kontrast des LCD-Moduls regeln

Befehlsstruktur
CONT [Kontrastwert]

Erklärung des Befehls
Das LCD-Modul ist hinterleuchtet. Zeichen werden schwarz dargestellt. Mit dem CONT-Befehl kann die Stärke der Farbe eingestellt werden. Numerale, Konstanten und Variablen können für den [Kontrastwert] genutzt werden. Je größer der [Kontrastwert] ist, desto dunkler werden die Zeichen. Der Startwert ist 7.

Beispiel des Befehls
LCDINIT

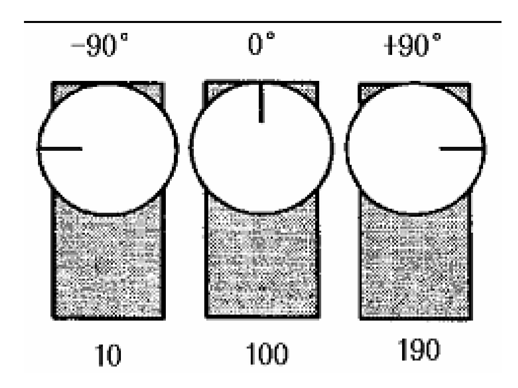
CONT 10
PRINT "robonova"

Kapitel 8

Erklärung der Motor-

steuerbefehle in roboBASIC

Die Roboter-Kontrolleinheit kann Servomotoren und Gleichstrommotoren steuern. Bei Gleichstrommotoren kann die Kontrolleinheit Geschwindigkeit, Richtung und das Anhalten des Motors durch die Benutzung von digitalen Ein- und Ausgabebefehlen steuern.
Der Arbeitsbereich eines Servomotors reicht von -90° bis +90°. Bei der Benutzung von Schrittmotoren in roboBASIC werden Gradzahlen in Form von Numeralen zwischen 10 und 190 genutzt, weil die Kontrolleinheit keine negativen Werte verarbeitet.



Verschiedene Servomotoren:



HS-311 HS-5645MG HSR-8498HB Schrittmotor

ZERO

Nullpunkt eines Servo-Motors

Befehlsstruktur:

MR-C2000-Serie:

ZERO [Standardpunkt von Motor 0], [Standardpunkt von Motor 1], …, [Standardpunkt von Motor 5]

MR-C3000-Serie:

ZERO [Gruppen-Nullpunkt], [Standardpunkt von Motor n], …

Erklärung des Befehls:

Der Nullpunkt eines Servomotors ist vom Motor selbst abhängig. Das liegt an der Fertigungstoleranz der Motoren. Einige Nullpunkte können 99 oder 98, andere 101 oder 102 sein. Diese Art von Fehlern können durch den ZERO-Befehl ausgeglichen werden. Wenn der Nullpunkt für einen Servomotor einmal gesetzt wurde ist er der Standardpunkt für den MOVE-Befehl.
Der Nullpunkt wird im EEPROM gespeichert um einen Verlust beim Ausschalten zu vermeiden.

Nullpunkt bei Kontrolleinheiten der CR-2000-Serie:

1. Alle Nullpunkte für alle Servomotoren zum Löschen alter Daten auf 100 setzen
2. Motor-Drehrichtung auf die Standardrichtung stellen
3. Ausschalten und wieder Einschalten
4. Motor mit Hilfe der Online-Funktion in die Nullstellung bewegen
5. Diese Position als Nullstellung mit Hilfe des ZERO-Befehls speichern

Beispiel 1:
Alten Nullpunkt löschen

```
ZERO 100,100,100,100,100,100
DIR 1,1,1,1,1,1                    Motordrehrichtung in die normale Richtung setzen
```

Beispiel 2:
Nullpunkt erneut einstellen und abspeichern

```
DIR 1,1,1,1,1,1
ZERO 100, 101, 99                  Nullpunkt von 3 Servomotoren setzen
ZERO 102, 100, 100, 99, 101, 100   Nullpunkt von 6 Servomotoren setzen
```

Bei der MR-C2000-Serie muss der Nullpunkt eine Gradangabe zwischen 90 und 100 sein.

Nullpunkt bei Kontrolleinheiten der CR-3000-Serie:
Beim Setzen des Nullpunktes bei Kontrolleinheiten der C-3000-Serie muss eine Gruppe deklariert werden.

```
ZERO G8B, 80, 120, 115, 80, 117, 88, 95, 120   Den Nullpunkt der Gruppe 8B setzen
                                               (Servomotoren #8 ~ #16)
```

Bei der MR-C3000-Serie muss der Nullpunkt eine Gradangabe zwischen 80 und 120 sein.

MOTOR

Die Nutzung des Servomotors einstellen

Befehlsstruktur:
MR-C2000-Serie:
MOTOR [Motornummer]
MR-C3000-Serie:
MOTOR [Motornummer] / [Spezielle Gruppe]

Erklärung des Befehls:

Servomotor in der C-2000-Serie einrichten:
Bei Kontrolleinheiten der C-2000-Serie gibt es 6 Ports (#0 ~ #5) für Servomotoren. Die Motorzahlen können mit 0 ~ 5 angegeben werden. Wenn alle Motoren genutzt werden sollen wird die Zahl 6 eingesetzt.

Ohne Nennung der Motorennummer funktionieren die Motoren nicht. Bei der [Motornummer] können nur Zahlen von 0 bis 6 genutzt werden.

Beispiel 1:

MOTOR 6	Alle Motoren (#0 – #5) werden genutzt

Beispiel 2:

MOTOR 2	Motor #2 wird genutzt

Servomotor in der C-3000-Serie einrichten:
Bei der C-3000-Serie gibt es 32 Ports für 32 Servomotoren. Jeder Motor kann einzeln mit [Motornummer] angesprochen werden. Gruppen können mit [speziellen Gruppen] angesprochen werden.
Für [Motornummer] können Numerale, Byte-Variablen oder Konstanten genutzt werden.

Beispiel des Befehls:

Beispiel 1:	MOTOR 0	Servomotor 0 wird genutzt
Beispiel 2:	MOTOR G6A	Servomotorgruppe 6A (#0~#5) wird genutzt
	MOTOR G6C	Servomotorgruppe 6C (#12~#17) wird genutzt
Beispiel 3:	MOTOR G8A	Servomotorgruppe 8A (#0~#7) wird genutzt
Beispiel 4:	Servomotor mit einer Variable setzen	
	DIM I AS BYTE	
	FOR I = 0 TO 31	Servomotoren (#0~#31) werden durch die Variable I genutzt
	MOTOR I	
	NEXT I	
Beispiel 5:	MOTOR G24	Servomotorgruppe 24 (#0~#23) wird genutzt
Beispiel 6:	MOTOR ALLON	Alle Servomotoren werden genutzt

MOTOROFF
Servomotor ausschalten

Befehlsstruktur:
MR-C2000-Serie:
MOTOROFF [Motornummer]

MR-C3000-Serie:
MOTOROFF [Motornummer] / [spezielle Gruppe]

Erklärung des Befehls:
Der MOTOROFF-Befehl ist genau das Gleiche wie der MOTOR-Befehl.

MOVE
Mehrere Servomotoren zur gleichen Zeit benutzen

Befehlsstruktur:
MR-C2000-Serie:
MOVE [Winkel von Motor 0], [Winkel von Motor 1], …, [Winkel von Motor 6]

MR-C3000-Serie:
MOVE [spezielle Gruppe], [Winkel von Motor n]

Erklärung des Befehls:
MOVE-Befehl bei der MR-C2000-Serie:
Der MOVE-Befehl bewegt den Servomotor zu einem bestimmten Winkel. Bei der Benutzung wird die PWM-Funktion deaktiviert. Der Bereich der [Motorenwinkel] liegt zwischen 10 und 190.
Wenn die Servomotoren #1, #3 und #4 benutzt werden sollen sieht der Befehl so aus:
MOVE 60, 100, 120

Wenn nur der Motor #2 genutzt werden soll sieht der Befehl folgendermaßen aus:
MOVE , 140

Dieser Prozess stellt sich nun aber sehr komplex dar, vor allem, wenn 6 Motoren zur gleichen Zeit bewegt werden müssen. Der Prozess wir einfachen, wenn die „Servomotor-Echtzeit-Kontrolle" genutzt wird.
Wenn ein Gleichstrommotor genutzt wird heißt 100 Stop und 190 maximale Geschwindigkeit mit Zählrotation und 10 heißt maximale Geschwindigkeit mit normaler Rotation.

Wenn der Motor nach einer vorherigen Operation genutzt wird sollte der WAIT-Befehl eingesetzt werden.

Beispiel des Befehls:
MOVE 100, 50, 140, 120, 80, 40
MOVE 120, , , 160
MOVE , 70, 100
MOVE , , , , , 100

MOVE-Befehl bei der MR-C3000-Serie:
Bei Kontrollern der MR-C3000-Serie sind die Ports für Servo- und Gleichstrommotoren verschieden. Deshalb kann der MOVE- und der PWM-Befehl gleichzeitig genutzt werden.

Beispiel des Befehls:
Beispiel 1:
MOVE G6A, 85, 113, 72, 117, 115, 100
MOVE G6C, 75, , 96, 123, , 122
MOVE G8A, 85, 113, 72, 117, 115, 100, 95, 45

Beispiel 2:
MOVE G24, 85, 113, 72, 117, 115, 100
Ist das gleiche wie:
MOVE24 85, 113, 72, 117, 115, 100

SPEED

Geschwindigkeit eines Servomotors einstellen

Befehlsstruktur:

SPEED [Motorgeschwindigkeit]

Erklärung des Befehls:

Der SPEED-Befehl stellt die Geschwindigkeit von Servomotoren bei der Benutzung des MOVE-Befehls ein.
Bei der MR-C2000-Serie können Numerale oder Konstanten zwischen 1 ~ 15 für die [Motorgeschwindigkeit] genutzt werden.
Bei der MR-C3000-Serie können Byte-Variablen genutzt werden.
Die Standardeinstellung ist 3. Eine zu große Geschwindigkeit ist für den Nutzer genauso gefährlich wie für den Roboter.

Beispiel des Befehls:

Beispiel 1:

```
SPEED 7                  Motorgeschwindigkeit mit 7 definieren
```

Beispiel 2:

```
DIM STEP_SPEED AS BYTE        STEP_SPEED als Variable deklarieren
STEP_SPEED = 15               STEP_SPEED den Wert 15 zuweisen
SPEED STEP_SPEED              SPEED den Wert von STEP_SPEED zuweisen
```

ACCEL

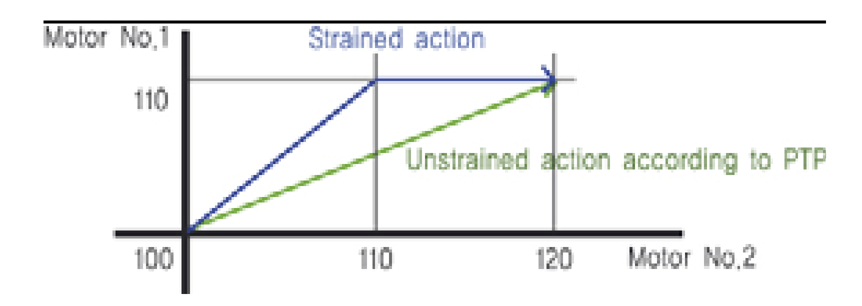
Die Beschleunigung eines Servomotors einstellen

Befehlsstruktur:

ACCEL [Motorbeschleunigung]

Erklärung des Befehls:

Der ACCEL-Befehl verändert die Servo-Geschwindigkeit von 0 bis auf den angegebenen Wert.



Für die [Motorbeschleunigung] können Numerale oder Konstanten zwischen 0 und 15 genutzt werden. Der normale Wert ist 3. Eine größere Zahl erhöht die Beschleunigung des Servos.
Bei der ersten Benutzung eines Servomotors rotiert dieser sehr schnell zu der gewünschten Stellung. Um diese Anfangsgeschwindigkeit zu reduzieren ist es günstiger den ACCEL und SPEED-Befehl zu nutzen.
Bei der MR-C3000-Serie kann der ACCEL-Befehl nicht genutzt werden.

Beispiel des Befehls:
ACCEL 7 Beschleunigung des Servomotors mit 7 beschreiben

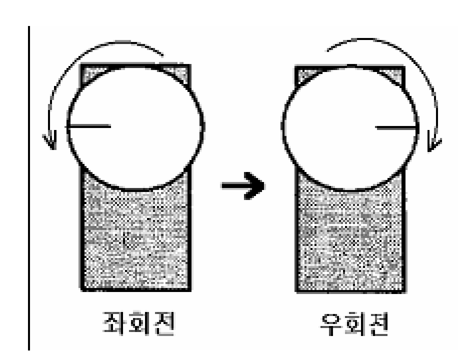
DIR

Die Bewegungsrichtung eines Servomotors einstellen

Befehlsstruktur:
MR-C2000-Serie:
DIR [Bewegungsrichtung Motor 0], [Bewegungsrichtung Motor 1], ..., [Bewegungsrichtung Motor 5]

MR-C3000-Serie:
DIR [spezielle Gruppe], [Bewegungsrichtung von Motor n], …

Erklärung des Befehls:
Ein Stellmotor dreht sich nach links, wenn der Winkelsatz kleiner als 100 ist (Standardwinkel) und dreht sich nach rechts, wenn der Winkelsatz größer als 100 (Standardwinkel).
Im unteren Bild beträgt die Rotation nach links (Standardrichtung) 10 Grad.



Für die [Bewegungsrichtung] können Konstanten oder Numerale wie 0 (Zählrotation/links) oder 1 (normale Rotation/rechts) genutzt werden. Der Standardwert ist 0. Wenn zum Beispiel 4 Servomotoren genutzt werden und der Motor #3 explizit erwähnt wird (DIR 0, 1, , 0) so dreht er sich in die Standardrichtung.

Beispiel des Befehls:
Beispiel zur MR-C2000-Serie:
DIR 0, 1, 1, 0, 1, 0
DIR , , 0

Beispiel zur MR-C3000-Serie:
DIR G8A, 0, 1, 0, 0, 1, 0, 0, 0
DIR G8B, 1, 0, 1, 1, 0, 1, 1, 1

PTP **Punkt zu Punkt**

Einstellung der Ein/Aus-Funktion für die gleichzeitige Kontroller mehrerer Servomotoren

Befehlsstruktur:
MR-C2000-Serie:
PTP [Startwert]

MR-C3000-Serie:
PTP [SETON/SETOFF/ALLON/ALLOFF]

Erklärung des Befehls:
Im Falle von mehreren Bewegungen und Bewegungen zu verschiedenen Winkeln unterscheiden sich die Endzeiten der Motoren untereinander. So kann es bei Robotern mit Armen und Robotern mit Servos zu instabilen Bewegungen kommen.
In der Roboterentwicklung gibt es eine Möglichkeit die Zeiten bei Erreichen der Endpunkte von Servomotoren zu berechnen und somit alle Bewegungen gleichzeitig zu beenden um einen geschmeidigen Bewegungsablauf zu ermöglichen.
Die Kontrolleinheiten der MR-Serie können diese Punkt-zu-Punkt-Methode nutzen.
Dafür wird der PTP-Befehl verwendet.

PTP-Kontrolle bei der MR-C2000-Serie:
Für den [Startwert] kann 0 (abbrechen) oder 1 (Konfiguration) in Numeralen oder Konstanten verwendet werden. Bei der Benutzung von 2 Servomotoren (Nr. 1 und Nr. 2) sieht das dann so aus:

Beispiel einer theatralischen Bewegung:

```
PTP 0
MOVE 100, 100
MOVE 110, 120
```

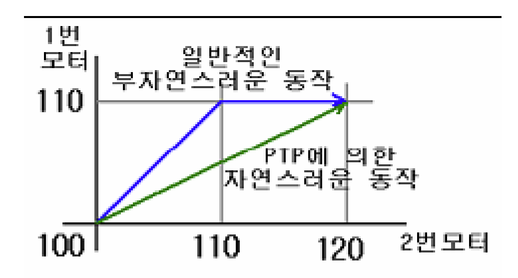
Beschreibung: Motor 1 bewegt sich um 10° und Motor 2 um 20°. Beide Motoren bewegen sich mit der gleichen Geschwindigkeit. Sobald der erste Motor seine 10 Grad erreicht hat stoppt er und Motor 2 bewegt sich für die restlichen 10 Grad alleine weiter.

Beispiel einer geschmeidigen Bewegung:

```
PTP 0
MOVE 100, 100
MOVE 110, 120
```

Beschreibung: Motor 1 bewegt sich um 10° und Motor 2 um 20°, allerdings bewegt sich Motor 1 mit der halben Geschwindigkeit. Beide Motoren stoppen daher zur selben Zeit.

Beachten Sie das folgende Diagramm zum Vergleich der Benutzung des PTP-Befehls (grüne Linie) oder der Nichtbenutzung des PTP-Befehls (blaue Linie).



Beachten Sie, dass bei normaler Bewegung sich die Motoren bis zu einem Winkel von 110 Grad gleichzeitig bewegen und sich danach nur Motor 2 bis 120 Grad weiterbewegt.
Immer dann, wenn der PTP-Befehl genutzt wird berechnet die Kontrolleinheit die Bewegung und beide Motoren beenden die Bewegung zur gleichen Zeit.

PTP-Kontrolle bei der MR-C3000-Serie:
Bei der MR-C3000-Serie können mehrere Servomotoren genutzt werden. Die Funktion PTP kann dazu genutzt werden alle Servomotoren oder individuelle Gruppen zu steuern.

PTP SETON(PTP setup)	PTP-Funktion zur Gruppennutzung einstellen
PTP SETOFF(PTP cancel)	PTP-Funktion zur Gruppennutzung abbrechen
PTP ALLON(PTP all setup)	PTP-Funktion zur Nutzung aller Motoren einstellen
PTP ALLOFF(PTP all cancel)	PTP-Funktion zur Nutzung aller Motoren abbrechen

Bei der MR-C3000-Serie muss der Befehl WAIT am Ende der Bewegung für jede Gruppe genutzt werden, damit die Bewegung der Motoren zur gleichen Zeit endet.

SERVO
Ein Servomotor wird benutzt

Befehlsstruktur:
SERVO [Motornummer],[Motorwinkel]

Erklärung des Befehls:
Dieser Befehl stellt den gewünschten Winkel eines Motors ein. Beim MR-C2000-Kontroller wird die PWM-Funktion abgebrochen. Die [Motornummer] ist der Port des Motors an der Kontrolleinheit. Der [Motorwinkel] muss zwischen 10 und 190 Grad liegen. Es ist die Nutzung von Numeralen, Konstanten oder Byte-Variablen möglich.

Beispiel des Befehls:
Beispiel 1:

```
SERVO 1,130        Motor Nummer 1 an die Position 130 bewegen
```

Beispiel 2:

```
DIM I AS BYTE

FOR I = 10 TO 190
    SERVO 4, I
    DELAY 100
    NEXT I
```

PWM   Pulsbreite
Kontrolle

Befehlsstruktur:
PWM [Motornummer],[Pulsbreitenwert]

Erklärung des Befehls:
MR-C2000-Serie:
Servomotorsteuerports und PWM-Ports werden bei der MR-C2000-Serie in Verbindung genutzt. Die möglichen [Motornummern] sind 0 bis 5.
Wenn der PWM-Befehl mit einer Pulsbreite ausgeführt wird führt das zu einem Abbruch der Servofunktion.
Warnung: Ein einem Programm können der SERVO oder MOVE-Befehl und der PWM-Befehl nicht zusammen genutzt werden)

Geschwindigkeit	Pulsbreitenwert	Geschwindigkeit	Pulsbreitenwert
0	0	***60***	153
10	25	***70***	178
20	51	***80***	204
30	77	***90***	229
40	102	***100***	254
50	127		

PWM 3, 127 PWM-Ausgabe mit 50% Geschwindigkeit am Motorport 3

MR-C3000-Serie:
Die Servomotorsteuerung und die PWM-Funktion nutzen nicht dieselben Ports. Die MR-C3000-Serie besitzt 3 PWM-Ports (0-3) mit einer PWM-Frequenz von 454Hz.

PWM 0, 120 Pulsausgabe mit einer Geschwindigkeit von 120 am Port 0

FASTSERVO
Einen Servomotor mit höherer Geschwindigkeit betreiben

Befehlsstruktur:
FASTSERVO [Motornummer], [Motorwinkel]

Erklärung des Befehls:
Dieser Befehl bewegt einen Servomotor so schnell wie möglich zu einem gewünschten Winkel.
Nur Kontrolleinheiten der MR-C2000-Serie:
Die [Motornummer] ist der Port, an dem der Motor angeschlossen ist. Der Wert des [Motorwinkel] ist numerisch oder konstant und liegt zwischen 10 und 190.

Beispiel des Befehls:
FASTSERVO 2, 190 Motor 2 zum Winkel 190 so schnell wie möglich bewegen

HIGHSPEED
Die Bewegung eines Servomotors auf den Hochgeschwindigkeitsmodus schalten

Befehlsstruktur:
HIGHSPEED [SETON/SETOFF]

Erklärung des Befehls:
Einstellen oder Abbrechen der Hochgeschwindigkeitsmodus bei Kontrolleinheiten der MR-C3000-Serie. Der Hochgeschwindigkeitsmodus ist etwa dreimal so schnell wie die normale Geschwindigkeit.

HIGHSPEED SETON: Alle Servomotoren auf den Hochgeschwindigkeitsmodus bei Kontrolleinheiten der MR-C3000-Serie setzen
HIGHSPEED SETOFF: Abbruch des Hochgeschwindigkeitsmodus bei der MR-C3000-Serie (Rückkehr zum Standardmodus)

Beispiel des Befehls:

```
HIGHSPEED SETON                 Einstellen des Hochgeschwindigkeitsmodus
```

MOVEPOS — Bewegungsposition
POS — Motorposition

Einstellen der Bewegungsposition oder der Motorposition

Befehlsstruktur:

```
MOVEPOS [Sprungmarke]
        ………….
        [Sprungmarke]: POS [Gruppenbezeichnung], [Motorwinkel], …
        ………….
        [Sprungmarke]:
        ………….
        [Sprungmarke]:
```

Erklärung des Befehls:
Wenn die Position des Roboters (besteht aus dem Befehl MOVE) bei Kontrolleinheiten der MR-C3000-Serie von anderen Befehlen mitgebracht wird, so wird sie mit dem Befehl POS [Motorposition] bearbeitet und die Sprungmarken mit dem Befehl MOVEPOS. Mit den Befehlen MOVEPOS und POS vereinfacht sich die Überarbeitung von roboBASIC-Programmen.

Beispiel des Befehls:

```
…………………………
MOVEPOS POS01                   MOVE-Befehl POS ist Teil von POS01
…………………………
POS01: POS G6A, 10, 32, 15, 120, 78, 93
POS02: POS G6A, 67, 47, 32, 153, 23, 33
POS03: POS G6A, 34, 37, 122, 162, 84, 28
```

FPWM — Frequentierter Puls

Ausgabe eines PWM-Signals mit variabler Frequenz

Befehlsstruktur:
PWM [Port], [Frequenz], [Geschwindigkeit]

Erklärung des Befehls:
Die Frequenz des ausgehenden PWM-Pulses wird bei Kontrolleinheiten der MR-C3000-Serie verändert.
Ports: 0 bis 2
Frequenz: 1 (kleine Frequenz) bis 5 (hohe Frequenz)
Geschwindigkeit: 0 bis 255

Beispiel des Befehls:
FPWM 0, 1, 127 — Ausgabe eines PWM-Signals mit 50% Geschwindigkeit und einer kleinen Frequenz am PWM-Port 0 bei Kontrolleinheiten der MR-C3000-Serie

MOVE24
Alle 24 Servomotoren bewegen

Befehlsstruktur:
MOVE24 [Motorwinkel von Motor 0], ………., [Motorwinkel von Motor 23]

Erklärung des Befehls:
Gruppen von Servomotoren (spezielle Gruppen) können bei der MR-C3000-Serie mit dem Befehl MOVE gesteuert werden. Die Befehle MOVE24 und MOVE G24 sind für die gleichzeitige Nutzung von 24 Servomotoren da. Diese Befehle sind zur Steuerung von Robotern mit 16 bis 24 Servos gut geeignet.

Beispiel des Befehls:
MOVE24 100, 45, 67, 44, 132, 122, , , , 76, 81, 90

INIT Initial
Die Grundposition des Roboters festlegen

Befehlsstruktur:
INIT [bestimmte Gruppe], [Motorwinkel von Motor n], …

Erklärung des Befehls:
Wenn eine Kontrolleinheit der MR-C3000-Serie in einem Roboter verbaut ist werden alle Motoren beim einschalten automatisch auf die Position 100 gefahren. Dies kann zu Beschädigungen am Roboter führen.
Um solche Beschädigungen auszuschließen ist es möglich die Grundposition der Motoren beim Einschalten auf eine andere Position als 100 zu setzen.
Im Falle eines analogen Servo-Motors (HS-Serie) wird der Befehl INIT verwendet.
Im Falle eines digitalen Roboterservomotors (HSR-Serie) wird der Befehl GETMOTORSET verwendet.

Beispiel des Befehls:
INIT G8A, 100, 45, 67, 44, 132, 122, 76, 81

MOTORIN() Motoreingabe()

Auslesen der aktuellen Geschwindigkeit eines Servomotors

Befehlsstruktur:
MOTORIN ([Motornummer])

Erklärung des Befehls:
Mit diesem Befehl ist es möglich den aktuellen Positionswert von jedem Roboterservomotor (HSR-Serie), welcher an einen MR-C3000-Controller angeschlossen ist auszulesen.
Für die [Motornummer] können die Numerale 0 bis 30 und Konstanten verwendet werden.
In Verbindung mit der Kontrolleinheit können Motorwinkel von 10 bis 190 Grad gelesen werden. Wenn kein Motor an der Kontrolleinheit angeschlossen ist wird 0 ausgelesen.

Beispiel des Befehls:
```
DIM S0 AS BYTE

MOTOR 0                     Motor 0 benutzen
S0 = MOTORIN(0)             Wert von Motor 0 in die Variable S0 schreiben
```

AIMOTOR AI-Motor-Einstellung

Benutzung eines AI-Motors

Befehlsstruktur:
AIMOTOR SETON/SETOFF/INIT/[Motornummer]/[spezielle Gruppe]

Erklärung des Befehls:
Der AI-Motor für Roboter wird von Megarobotics hergestellt. Ein Mikrokontrollerschaltkreis ist im AI-Motor eingebaut, welcher dann mit der Kontrolleinheit der MR-Serie via RS232 kommuniziert. Der AI-Motor wird von Kontrolleinheiten der MR-Serie genauso behandelt wir ein normaler Servomotor.

- Kontrolle des Winkels des Motors
- Derzeitige Position des Motors und Drehmoment durch die PDI-Kontrolle (PGAIN, DGAIN)

AI-Motoren können an die Ports 0 ~ 30 angeschlossen werden (31 Ports)
[Motornummer] bezeichnet den Motor.
[spezielle Gruppe] spricht eine Gruppe von Motoren an

Der Befehlsprozess verläuft ähnlich dem von MOTOR. Numerale, Konstanten oder Variablen vom Byte-Typ können für [Motornummer] genutzt werden.
Bei der Benutzung eines AI-Motors ist die Bezeichnung dessen als AI-Motor notwendig, wird bei allen anderen Servomotoren aber nicht benötigt.

- AIMOTOR SETON — Nutzung eines AI-Motors
- AIMOTOR SETOFF — Abbrechen der Nutzung eines AI-Motors
- AIMOTOR INIT — Bewegung des AI-Motors in seine Grundposition

Beispiel des Befehls:

AIMOTOR INIT	AI-Motor initialisieren
AIMOTOR SETON	Nutzung des AI-Motors festlegen
AIMOTOR 0	Motor 0 nutzen
AIMOTOR G6B	Gruppe 6B nutzen (Motoren 6-11)

AIMOTOROFF — AI-Motor abbrechen

Nutzung eines AI-Motors abbrechen

Befehlsstruktur:
AIMOTOROFF [Motornummer] / [spezielle Gruppe]

Erklärung des Befehls:
Den Befehl AIMOTOR abbrechen.
Der Befehl hat die gleiche Funktion wie MOTOROFF.

Beispiel des Befehls:

AIMOTOROFF 0	AIMOTOR an Kanal 0 abbrechen
AIMOTOROFF G6B	Alle Motoren der Gruppe 6B (Ports 6-11) stoppen
AIMOTOR SETOFF	SETOFF deklarieren

AIMOTORIN() — AI-Motor-Eingabe

Den aktuellen Wert eines AI-Motors auslesen

Befehlsstruktur:
AIMOTORIN ([Motornummer])

Erklärung des Befehls:
Mit diesem Befehl kann man den aktuellen Positionswert eines AI-Motors an einer Kontrolleinheit der MR-C3000-Serie einlesen.
Für [Motornummer] können Numerale von 0 bis 30 und Konstanten genutzt werden.
In Verbindung mit der Kontrolleinheit können Motorwinkel von 10 bis 190 Grad gelesen werden. Wenn kein Motor an der Kontrolleinheit angeschlossen ist wird 0 ausgelesen.

Beispiel des Befehls:

```
DIM AI5 AS BYTE

AIMOTOR INIT
AIMOTOR SETON
AIMOTOR 5               Nutzung des AI-Motors 5
AI5 = AIMOTORIN(5)      Der Variable AI5 den Wert von Motor 5 zuweisen
```

GETMOTORSET — Motoreingang konfigurieren

Die aktuelle Geschwindigkeit eines Motors auslesen und den Status verwalten

Befehlsstruktur:
GETMOTORSET [spezielle Gruppe], [Motor n der Gruppe], …

Erklärung des Befehls:
Durch ein interaktives Kommunikationssystem ist es möglich den aktuellen Positionswert eines digitalen Roboter-Servomotors (HSR-Serie), der an eine Kontrolleinheit der MR-C3000-Serie angeschlossen ist, auszulesen.
Wenn eine Kontrolleinheit der MR-C3000-Serie in einem Roboter verbaut ist werden alle Motoren beim einschalten automatisch auf die Position 100 gefahren. Dies kann zu Beschädigungen am Roboter führen.
Um solche Beschädigungen auszuschließen ist es möglich die Grundposition der Motoren beim Einschalten auf eine andere Position als 100 zu setzen.
Zuerst wir die aktuelle Position vor dem endgültigen Einschalten ausgelesen und dann wird die entsprechende Positionsänderung durch den MOVE-Befehl eingeleitet.

Bei dem [Motor n der Gruppe] kann 0 oder 1 genutzt werden. Im Falle von 1 wird der aktuelle Wert des ausgewählten Motors ausgelesen und spiegelt dann den aktuellen Zustand des Roboters wieder. Im Falle von 0 wird der Motor in die Standardposition von 0 bewegt.

Beispiel des Befehls:
GETMOTORSET G8A, 1, 1, 1, 1, 0, 0, 0, 0
Die Motoren 0, 1, 2 und 3 behalten die aktuelle Position bei
Die Motoren 4, 5, 6 und 7 bewegen sich auf die Position 100

Kapitel 9

roboBASIC

Befehle zur Musiksteuerung

Die Kontrolleinheiten der MR-Serie sind in der Lage Warntöne und Klänge wiederzugeben. Um diese Funktion nutzen zu können wird noch ein einfacher Piezo benötigt. Die Anschlüsse des Piezos beschränken sich auf einen roten (+) und einen weißen (-) Pin.
Wenn lauter und klarer Sound benötigt wird, allerdings sind dann aktive Lautsprecher oder ein Verstärker notwendig.

Verbindung mit Kontrolleinheiten der MR-C2000-Serie:
Der Piezo wird an den Port 8 einer MR-C2000-Kontrolleinheit angeschlossen. Der + Anschluss des Piezos wird mit VCC und der – Anschluss mit SIG vom Port 8 der Kontrolleinheit verbunden.

Verbindung mit Kontrolleinheiten der MR-C3000-Serie:
Der Piezo wird an den Port 28 einer MR-C3000-Kontrolleinheit angeschlossen. Der + Anschluss des Piezos wird mit VCC und der – Anschluss mit SIG vom Port 288 der Kontrolleinheit verbunden.

Bemerkung: Die MR-C3024-Kontrolleinheiten haben einen eingebauten Piezo.

BEEP

Einen Warnton mit einem Piezo ausgeben.

Befehlsstruktur:
BEEP

Erklärung des Befehls:
Um einen Warnton mit Kontrolleinheiten der MR-C2000-Serie zu erzeugen wird der Befehl BEEP genutzt. Wenn nur ein Summer verwendet werden soll, kann dieser ebenfalls an jedem anderen Ausgangsport angeschlossen werden. Ein einfaches Geräusch wird dann mit dem Befehl OUT ausgegeben.
Die Kontrolleinheiten der MR-C3000-Serie nutzen immer einen Summer mit dem Befehl OUT.

Beispiel des Befehls:
BEEP Ausgabe eines Warntons

SOUND

Einen Sound mit einem Piezo ausgeben.

Befehlsstruktur:
SOUND [Tonhöhe], [Länge], [Tonhöhe], [Länge], …

Erklärung des Befehls:
Bei Kontrolleinheiten der MR-C2000-Serie kann die Frequenz und die Dauer eines Tones eingestellt werden.
Konfigurationswerte: Von 1 bis 254
Die Belegung sieht so aus, wie in der Tabelle gezeigt:

Eingabe	Frequenz (Hz)	Eingabe	Frequenz (Hz)	Eingabe	Frequenz (Hz)
1	38,86k	***70***	800	***160***	389
2	23,81k	***80***	775	***170***	365
5	11,11k	***90***	689	***180***	344
10	5,88k	***100***	621	***190***	327
20	3,00k	***110***	565	***200***	311
30	2,00k	***120***	518	***210***	295
40	1,54k	***130***	478	***220***	283
50	1,23k	***140***	444	***230***	270
60	1,00k	***150***	413	***240***	260

Zeit	Länge (11ms)
0,5s	45
1s	90
2s	180

Beispiel des Befehls:

SOUND 60, 90, 60, 180, 30, 45 Einen Ton von 1kHz für 1s, 2s und von 2kHz für 0,5s generieren

PLAY

Musik mit einem Piezo abspielen

Befehlsstruktur:

PLAY „[Notenfolge]"

Erklärung des Befehls:

Die roboBASIC-Sprache und die Kontrolleinheiten der MR-Serie besitzen die Möglichkeit Musik abzuspielen.
Um Musik abzuspielen müssen Daten zur [Notenfolge] hinzugefügt werden.
Siehe untenstehende Tabelle:

Notenfolge	Beschreibung
C	„Do"
D	„Re"
E	„Mi"
F	„Fa"
G	„Sol"
A	„La"
B	„Si"
T	Kontrolle des Tempos, Rhythmus und Geschwindigkeit
L	Tiefe Oktave wählen
M	Mittlere Oktave wählen
H	Obere Oktave wählen
#, +	Einen Ton erhöhen (#)
$, -	Einen Ton erniedrigen (♭)
P, ,(Rest von a)	Rest von A
<	Oktave fallen lassen
>	Oktave anheben

T heißt Tempo und das Standard-Tempo ist 7.
Das Tempo kann von 1 bis 0 eingestellt werden (0 heißt 10). Das schnellste Tempo ist somit 1 und 0 das langsamste.
Bei Kontrolleinheiten der MR-C2000-Serie können 3 Oktaven genutzt werden: Die niedrige, mittlere und die hohe Oktave.
Um die Länge eines Tones zu bestimmen werden die Numerale von 0 bis 9 verwendet. Dazu betrachten sie bitte folgende Tabelle:

Note	A Whole note	A Half note	A dotted Half note	A Quarter note	A dotted Quarter note	An eighth note	A dotted eighth note	A sixteenth note	A dotted sixteenth note	A thirty –second note
No.	1	2	3	4	5	8	9	6	7	0

Wenn die Länge eines Tones nicht angegeben wird, wird der Ton am längsten gespielt.
„4CDEF8G“ bedeutet, dass die Töne Do, RE, Mi, Fa also Viertelnoten und Sol als Achtel gespielt wird.
Der Standart des Befehls PLAY ist die mittlere Oktave, halbe Noten und ein Tempo von 7.

1.) Die Länge eines Tone hängt mit dem Ton zusammen (4Do 8Mi 6Fa)
2.) # oder andere Markierungen steh vor dem Ton (#Do $Mi +Fa –Sol 4#Do #4Do)

Bei Kontrolleinheiten der MR-C3000-Serie muss statt PLAY der Befehl MUSIC verwendet werden.

Beispiel des Befehls:
PLAY „M4GGAA GGE GGEED“
PLAY „M4GGAA GGE GEDEC“

MUSIC

Musik mit einem Piezo abspielen

Befehlsstruktur:
PLAY „[Notenfolge]“

Erklärung des Befehls:
Die roboBASIC-Sprache und die Kontrolleinheiten der MR-Serie besitzen die Möglichkeit Musik abzuspielen.
Um Musik abzuspielen müssen Daten zur [Notenfolge] hinzugefügt werden.
Siehe untenstehende Tabelle:

Notenfolge	Beschreibung
C	„Do“
D	„Re“
E	„Mi“
F	„Fa“
G	„Sol“
A	„La“
B	„Si“
[	Eine Gruppe von Tönen um das 1,5fache kürzen
]	Eine Gruppe von Tönen um das 1,5fache verlängern
O	Eine Oktave auswählen
M	3 Oktaven auswählen
.	Einen Ton um das 1,5fache verlängern
#, +	Einen Ton erhöhen (#)
$, -	Einen Ton erniedrigen (♭)
P, ,(Rest von a)	Rest von A
<, L	Oktave fallen lassen
>, H	Oktave anheben

Bei Kontrolleinheiten der MR-C3000-Serie können 7 Oktaven genutzt werden.

Um die Länge eines Tones zu bestimmen werden die Numerale von 0 bis 9 verwendet. Dazu betrachten sie bitte folgende Tabelle:

Im Falle punktierter Noten wird dies als Numeral oder . in der [Notenlinie] vermerkt.

Note	A Whole note	A Half note	A dotted Half note	A Quarter note	A dotted Quarter note	An eighth note	A dotted eighth note	A sixteenth note	A dotted sixteenth note	A thirty -second note
No.	1	2	3	4	5	8	9	6	7	0

Bei Kontrolleinheiten der MR-C2000-Serie muss statt MUSIC der Befehl PLAY verwendet werden.

Der Befehl TEMPO ist bei Kontrolleinheiten der MR-C3000-Serie extra verfügbar.

Beispiel des Befehls:
MUSIC „O34GGAA GGE GGEED“
MUSIC „O3GGA4.A GGE GEDEC“

TEMPO

Tempo der Musik konfigurieren

Befehlsstruktur:
TEMPO [Wert]

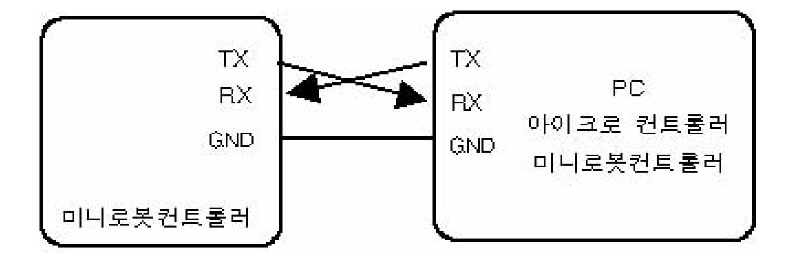
Erklärung des Befehls:
Der Befehl wird genutzt um das Tempo in Verbindung mit dem Befehl MUSIC bei Kontrolleinheiten der MR-C3000-Serie zu bestimmen.

Kapitel 10

roboBASIC

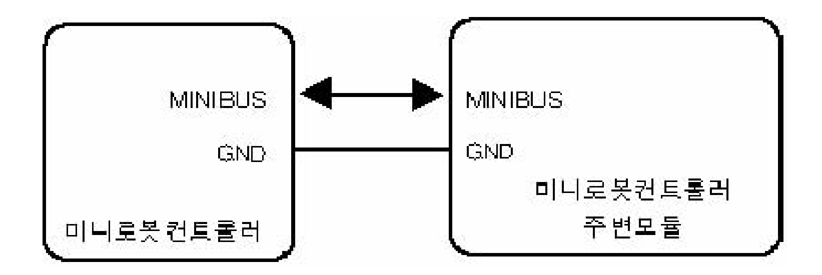
Befehle zur externen Kommunikation

Externe Kommunikation mit Kontrolleinheiten der MR-C2000-Serie:
Es gibt zwei externe Kommunikationsarten, die mit Kontrolleinheiten der MR-C2000-Serie genutzt werden können:
Die serielle RS232-Kommunikation und MiniBus.
Bei der seriellen Kommunikation ist eine Verbindung mit einem PC oder einer anderen MR-Kontrolleinheit möglich. Weiterhin kann die Kommunikation mit Kabel oder kabellos durch die Benutzung eines RF-Moduls erfolgen. Zur RS232-Kommunikation werden 3 Kabel benötigt (Senden – TX, Empfangen – RX, Masse – GND)



Zum Anschluss an einen PC ist eine Spannungsversorgung (MAX32 oder ähnliche) notwendig.

Beim MiniBus werden das BUS-Signal und Masse zur Kommunikation genutzt.



Der MiniBus nutzt nur ein Kabel zur Kommunikation. Allerdings sind strengere Regeln notwendig. Das LCD-Modul ist ein Beispiel der Übertragung von Daten von der Kontrolleinheit zum LCD via MiniBus.

Externe Kommunikation mit Kontrolleinheiten der MR-C3000-Serie:
Bei Kontrolleinheiten der MR-C3000-Serie kann die Hochgeschwindigkeits-RS232-Verbindung genutzt werden.
Allerdings ist eine MiniBus-Kommunikation mit Kontrolleinheiten der MR-C3000-Serie nicht möglich.

Beide Kontrolleinheiten haben eine maximale Verbindungsgeschwindigkeit von 115.200bps.

RX

RX-Port empfängt RS232-Signal

Befehlsstruktur:
RX [Portgeschwindigkeit], [Empfangsvariablen], [Fehlererkennungslabel]

Erklärung des Befehls:
Durch die Nutzung von Port 9 einer MR-C2000-Kontrolleinheit können Daten via RS232 empfangen werden.
Die [Portgeschwindigkeit] wird durch die Zahlen 1 bis 4 beschrieben, wobei jede Zahl für spezifische Porteigenschaften steht, welche im Folgenden erklärt sind:

Zahl	Porteigenschaften
1	1200bps, 8Bit Daten, keine Parität, 1 Stop bit
2	2400bps, 8Bit Daten, keine Parität, 1 Stop bit
3	2400bps, 8Bit Daten, keine Parität, 1 Stop bit
4	4800bps, 8Bit Daten, keine Parität, 1 Stop bit

[Empfangsvariablen] stellen die zu empfangenden Daten dar. Nur als Byte-Form deklarierte Variablen sind zugelassen.

Das [Fehlererkennungslabel] ist ein Label für Fehler, die während des Empfangs auftreten können. Wie in dem Falle, wenn der Empfangspuffer leer ist. Alle Programme, die Daten durch RS232 erwarten können folgende Befehlsstruktur nutzen:

Erneuter Versuch:

RX 4, A, Retry

Um RS232 Daten mit einer Kontrolleinheit der Serie MR-C3000 zu empfangen muss statt dem Befehl RX der Befehl ERX genutzt werden.

Beispiel des Befehls:
In diesem Beispiel wird ein ASCII-Code &8h80 (16 Antilogarithmus) via RS232 von einem externen Terminal empfangen. Die LED am Port 0 wird dann eingeschalten und alle anderen aus.

```
DIM A AS BYTE

Retry: RX 4, A, Retry
        IF A = &h80 THEN
                OUT 0, 0
        ELSE
                OUT 0, 1
        ENDIF
        GOTO Retry
```

TX

TX-Port sendet RS232-Signal

Befehlsstruktur:
TX [Portgeschwindigkeit], [Daten]

Erklärung des Befehls:
Durch die Nutzung von Port 10 einer MR-C2000-Kontrolleinheit können Daten via RS232 gesendet werden.
Die [Portgeschwindigkeit] wird durch die Zahlen 1 bis 4 beschrieben, wobei jede Zahl für spezifische Porteigenschaften steht, welche im Folgenden erklärt sind:

Zahl	Porteigenschaften
1	1200bps, 8Bit Daten, keine Parität, 1 Stop bit
2	2400bps, 8Bit Daten, keine Parität, 1 Stop bit
3	2400bps, 8Bit Daten, keine Parität, 1 Stop bit
4	4800bps, 8Bit Daten, keine Parität, 1 Stop bit

[Daten] sind die Datenwerte, welche durch den TX-Port gesendet werden sollen. Numerale, Konstanten und Variablen können genutzt werden. Wenn der Buchstabe A gesendet werden soll, sollte der entsprechende ASCII-Code für den Buchstaben A gesendet werden. Beachten sie das folgende Beispiel:

```
DIM I AS BYTE
I = "A"
TX 4, I
```

Sozusagen werden Buchstaben in ASCII umgewandelt und dann in die Variablen gespeichert.

Um RS232 Daten mit einer Kontrolleinheit der Serie MR-C3000 zu empfangen muss statt dem Befehl TX der Befehl ETX genutzt werden.

Beispiel des Befehls:
Im folgenden Beispiel wird ein Wert von 0 ständig via RS232 an ein externes Terminal gesendet.

```
DIM A AS BYTE

Main:
        A = BYTEIN(0)
        TX 4, A
        GOTO Main
```

MINIIN

Empfangen eines MiniBus-Signals am MiniBus-Port

Befehlsstruktur:
MINIIN

Erklärung des Befehls:
MiniBus-Daten werden durch einen der 6 MiniBus-Ports an der MR-C2000-Kontrolleinheit empfangen. Dieser Vorgang dauert so lange an bis 0 empfangen wird. Um Daten durch den MiniBus zu empfangen kann das Programm folgendermaßen geschrieben werden:

```
DIM A AS BYTE

Retry:
        A = MINIIN
        IF A = 0 THEN GOTO Retry
```

MINIOUT

Senden von MiniBus-Signalen durch den MiniBus

Befehlsstruktur:
MINIOUT [Daten], [Daten], …

Erklärung des Befehls:
MiniBus-Daten werden durch den MiniBus-Port 6 einer MR-C2000 Kontrolleinheit gesendet. MiniBus kann die gleichen Funktionen wie das RS232-Protokoll ausführen. Z.B.: 4800bps, 8bit Daten, keine Parität, 1Stop-Bit. Numerale, Konstanten und Variablen können in [Daten] genutzt werden. Weiterhin kann eine unbeschränkte Anzahl von Daten gesendet werden. Allerdings kann der Numeral 0 nicht gesendet werden.

Beispiel des Befehls:

```
MINIOUT 100, 20, 76, 65
```

ERX

RS232 Signale mit einem ERX-Port empfangen

Befehlsstruktur:
ERX [Portgeschwindigkeit], [Empfangsvariablen], [Fehlererkennungslabel]

Erklärung des Befehls:
Daten werden durch den ERX-Port bei Kontrolleinheiten der MR-C3000-Serie empfangen.
Die Konstanten für [Portgeschwindigkeit] sind unten aufgelistet:

Zahl	Porteinstellungen
2400	2400bps, 8Bit Daten, Keine Parität, 1 Stop bit
4800	4800bps, 8Bit Daten, Keine Parität, 1 Stop bit
9600	9600bps, 8Bit Daten, Keine Parität, 1 Stop bit
14400	14400bps, 8Bit Daten, Keine Parität, 1 Stop bit
19200	19200bps, 8Bit Daten, Keine Parität, 1 Stop bit
28800	28800bps, 8Bit Daten, Keine Parität, 1 Stop bit
38400	38400bps, 8Bit Daten, Keine Parität, 1 Stop bit
57600	57600bps, 8Bit Daten, Keine Parität, 1 Stop bit
76800	76800bps, 8Bit Daten, Keine Parität, 1 Stop bit
115200	115200bps, 8Bit Daten, Keine Parität, 1 Stop bit
230400	230400bps, 8Bit Daten, Keine Parität, 1 Stop bit

Für die [Empfangsvariablen], welche die zu sendenden Daten beinhalten, können nur Byte-Variablen genutzt werden.

Im [Fehlererkennungslabel] stehen die Befehle, die ausgeführt werden sollen, wenn keine Daten durch den RS232-Port empfangen werden.

```
Retry:
        ERX 9600, A, Retry
```

Um RS232 Daten mit einer Kontrolleinheit der Serie MR-C2000 zu senden muss statt dem Befehl ERX der Befehl RX genutzt werden.

ETX

RS232 Signale mit einem ERX-Port senden

Befehlsstruktur:
ETX [Portgeschwindigkeit], [Daten]

Erklärung des Befehls:
Daten werden durch den ETX-Port bei Kontrolleinheiten der MR-C3000-Serie gesendet. Die Konstanten für [Portgeschwindigkeit] sind unten aufgelistet:

Zahl	Porteinstellungen
2400	2400bps, 8Bit Daten, Keine Parität, 1 Stop bit
4800	4800bps, 8Bit Daten, Keine Parität, 1 Stop bit
9600	9600bps, 8Bit Daten, Keine Parität, 1 Stop bit
14400	14400bps, 8Bit Daten, Keine Parität, 1 Stop bit
19200	19200bps, 8Bit Daten, Keine Parität, 1 Stop bit
28800	28800bps, 8Bit Daten, Keine Parität, 1 Stop bit
38400	38400bps, 8Bit Daten, Keine Parität, 1 Stop bit
57600	57600bps, 8Bit Daten, Keine Parität, 1 Stop bit
76800	76800bps, 8Bit Daten, Keine Parität, 1 Stop bit
115200	115200bps, 8Bit Daten, Keine Parität, 1 Stop bit
230400	230400bps, 8Bit Daten, Keine Parität, 1 Stop bit

[Daten] sind die Datenwerte, welche durch den TX-Port gesendet werden sollen. Numerale, Konstanten und Variablen können genutzt werden. Beachten sie das folgende Beispiel:

```
DIM I AS BYTE

I = "A"
ETX 9600, I
```

Um den Buchstaben A zu übertragen muss der ASCII-Code für den Buchstaben A gesendet werden.

Um RS232 Daten mit einer Kontrolleinheit der Serie MR-C2000 zu empfangen muss statt dem Befehl ETX der Befehl TX genutzt werden.

Kapitel 11

roboBASIC

Analoge Signalprozesse

Befehlserklärungen

AD()

Umwandlung eines analogen Signals vom AD-Port in dein digitales Signal

Befehlsstruktur:
AD([AD-Port])

Erklärung des Befehls:
Bei Kontrolleinheiten der MR-C3000-Serie sind 8 AD-Transformation-Ports vorhanden. Diese werden von 0 bis 7 benannt. Die digitalen Ein- und Ausgangsports sind die Ports von 32 bis 39). AD-Ports können ein analoges Signal eines externen Sensors oder Gerätes in digitale Signale umwandeln. Bei den Ports 0 bis 7 können Konstanten oder Byte-Variablen für [AD-Port] genutzt werden.

Beispiel des Befehls:
Im folgenden Beispiel wird ein Wert auf einem LCD-Modul ausgegeben, nachdem ein analoges Signal vom AD-Port 1 verarbeitet wurde.

```
DIM a AS BYTE           Byte-Variable a deklarieren
LCDINIT                 LCD-Modul initialisieren
CLS                     Alle Daten vom LCD-Modul löschen
CSOFF                   Den Cursor auf dem LCD ausblenden
MAIN:                   Eine Sprungmarke MAIN definieren
a = AD (1)              Der Wert vom AD-Port 1 wird in die Variable a gespeichert
LOCATE 5,0              Den Cursor auf die Position 5, 0 bewegen.
PRINT FORMAT(a,DEC,2)   Der in a abgespeicherte Wert wird als Dezimalzahl mit 2 Stellen
                        ausgegeben
GOTO MAIN               zurück zu MAIN
```

REMOCON()

Liest Werte von einer Infrarotfernbedienung von AD-Port #7

Befehlsstruktur:
REMOCON ([Remocon(#)])

Erklärung des Befehls:
Der Wert, welcher auf der Infrarotfernbedienung gedrückt wird, wird durch den AD-Port #7 an Kontrolleinheiten der MR-C3000-Serie empfangen.

[Remocon(#)] wird ab der Nummer 0 bis zu einer festen Anzahl von remocon(#) genutzt. Die verfügbaren remocon(#) können je nach Version der Kontrolleinheit erweitert werden.
Um mehr Einzelheiten zu erfahren können sie sich auf die Beispiele in den nächsten Kapiteln beziehen.



Remocon(0) (Modellname: MR-IR1)

Beispiel des Befehls:

```
DIM a AS BYTE                Variable a deklarieren
MAIN:                        Sprungmarke zur ständigen Abfrage definieren
a = REMOCON(0)               Der Wert von remocon wird in die Variable a gespeichert
ON a GOTO MAIN,KEY1,KEY2,KEY3,KEY4     Zu MAIN springen sobald ein Wert
                                       existiert
GOTO MAIN                    Zu MAIN springen
END
KEY1:                        Ausführen, wenn der empfangene Wert 1 ist
        ...................
GOTO MAIN                     MAIN springen
KEY2:                        Ausführen, wenn der empfangene Wert 1 ist
        ...................
GOTO MAIN                    Zu MAIN springen
KEY3:                        Ausführen, wenn der empfangene Wert 1 ist
        ...................
GOTO MAIN                    Zu MAIN springen
KEY4:                        Ausführen, wenn der empfangene Wert 1 ist
        ...................
GOTO MAIN                    Zu MAIN springen
```

SONAR()

Liest die kalkulierten Distanzen eines Ultraschallsensors am Ultraschallport aus

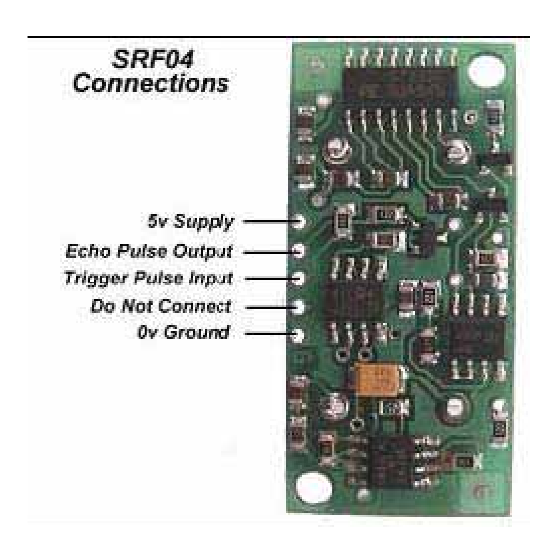
Befehlsstruktur:
SONAR([Ultraschallport])

Erklärung des Befehls:
Die digitalen Ein- und Ausgangsports der MR-C3000-Serie werden als Ultraschallports 0 bis 11 genutzt. Beachten sie Folgendes:

Digitaler Ein- und Ausgangsport der MR-C3000-Serie	Ultraschallport
Port #0	#0 Ultraschallausgangsport
Port #1	#0 Ultraschalleingangsport
Port #2	#1 Ultraschallausgangsport
Port #3	#1 Ultraschalleingangsport
Port #4	#2 Ultraschallausgangsport
Port #5	#2 Ultraschalleingangsport
Port #6	#3 Ultraschallausgangsport
Port #7	#3 Ultraschalleingangsport
Port #8	#4 Ultraschallausgangsport
Port #9	#4 Ultraschalleingangsport
Port #10	#5 Ultraschallausgangsport
Port #11	#5 Ultraschalleingangsport
Port #12	#6 Ultraschallausgangsport
Port #13	#6 Ultraschalleingangsport
Port #14	#7 Ultraschallausgangsport
Port #15	#7 Ultraschalleingangsport
Port #16	#8 Ultraschallausgangsport
Port #17	#8 Ultraschalleingangsport
Port #18	#9 Ultraschallausgangsport
Port #19	#9 Ultraschalleingangsport
Port #20	#10 Ultraschallausgangsport
Port #21	#10 Ultraschalleingangsport
Port #22	#11 Ultraschallausgangsport
Port #23	#11 Ultraschalleingangsport

Der Wert von SONAR muss eine feste Zahl sein. Ansonsten können Werte zwischen 0 und 3000 gewählt werden. Wenn der Wert 0 ist wird nicht gemessen, ansonsten ist ein mm-Wert die gemessene Distanz. Der verfügbare Ultraschallsensor für Kontrolleinheiten der MR-C3000-Serie ist das SRF04-Modell von der Firma ROBOT ELECTRONICS Inc.





Beispiel des Befehls:

DIM A AS INTEGER	Variable A als ganzzahlig deklarieren
A = SONAR(3)	Der gemessene Wert wird in Variable A gespeichert. Dabei wird der Ultraschallport #3 (Digitaler Ein- und Ausgansport #6 und #7) genutzt.

RCIN()

Die Werte von RC-Sendern und RC-Empfängern einlesen

Befehlsstruktur:
RCIN ([RC-Empfangsport])

Erklärung des Befehls:
RC-Empfänger werden an die AD-Eingansports einer MR-C3000-Kontrolleinheit angeschlossen. Auf diese Weise kann dann der Wert eines RC-Senders empfangen werden. Die folgende Tabelle zeigt die Konfigurationsmöglichkeiten des [RC-Empfangsport].

AD-Portnummer von Kontrolleinheiten der MR-C3000-Serie	RC-Empfangsport
Port #0 (Port #32)	RC-Empfangsport #0
Port #0 (Port #32)	RC-Empfangsport #1
Port #0 (Port #32)	RC-Empfangsport #2
Port #0 (Port #32)	RC-Empfangsport #3
Port #0 (Port #32)	RC-Empfangsport #4
Port #0 (Port #32)	RC-Empfangsport #5
Port #0 (Port #32)	RC-Empfangsport #6
Port #0 (Port #32)	RC-Empfangsport #7

Um Fehlfunktionen durch elektrische Interferenz zu vermeiden sollte FM anstatt von AM mit Kontrolleinheiten der MR-C3000-Serie genutzt werden. Wenn der Sender mehrere Funktionen hat, können mehrere Aktionen bzw. Bewegungen gesteuert werden.



Beispiel des Befehls:
DIM A AS BYTE

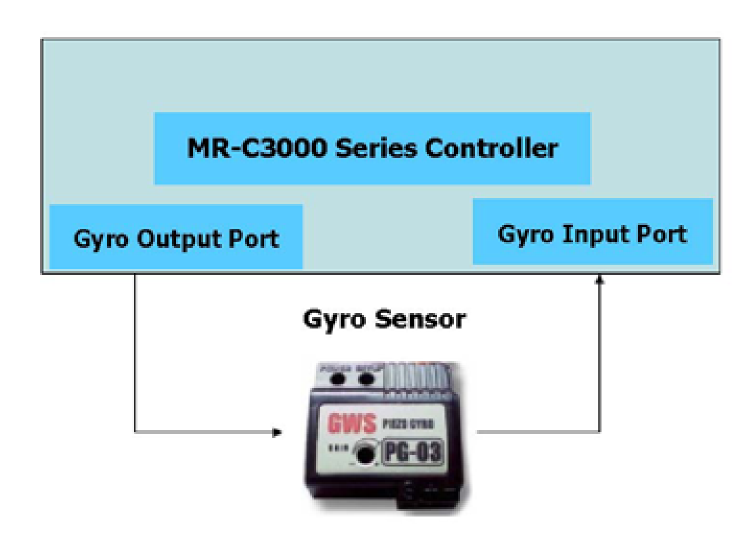
A = RCIN(0)   Ein empfangenes Funksignal wird in Variable A geschrieben

GYRODIR

Die Richtung eines Gyro-Sensors auslesen um einen Servomotor zu steuern

Befehlsstruktur:
GYRODIR [Gruppe], [Motorrichtung des Motor n] …

Erklärung des Befehls:
Wenn der aktuelle Wert des Gyro-Sensors auf die Servomotoren reflektiert wird setzt sich die Aktionsrichtung von selbst.
Es können bis zu 4 Gyrosensoren an einer MR-C3000-Kontrolleinheit angeschlossen werden.
Beachten sie Folgendes:



AD-Portnummer von Kontrolleinheiten der MR-C3000-Serie	Gyroport
Port #0 (Port #32)	Gyro #1 Kanal Ausgangsport
Port #1 (Port #33)	Gyro #2 Kanal Ausgangsport
Port #2 (Port #34)	Gyro #3 Kanal Ausgangsport
Port #3 (Port #35)	Gyro #4 Kanal Ausgangsport
Port #4 (Port #36)	Gyro #1 Kanal Eingangsport
Port #5 (Port #37)	Gyro #2 Kanal Eingangsport
Port #6 (Port #38)	Gyro #3 Kanal Eingangsport
Port #7 (Port #39)	Gyro #4 Kanal Eingangsport

Da Servos umkehrbar sind sollte die spezfische Richtung von dem aktuellen Wert des Servos bestimmt werden. Für die [Motorrichtung] sind die Werte 0 oder 1 zulässig. Ein Wert von 1 wird die Servostellung vergrößern, ein Wert von 0 verkleinert sie.

Beispiel des Befehls:
GYRODIR G6A, 1, 1, 0, 0, 1, 0

GYROSET

Benennt den Gyro-Sensor, welcher jeden angeschlossenen Servomotor beeinflusst

Befehlsstruktur:
GYROSET [Gruppe], [Motor n der Gruppe], …

Erklärung des Befehls:
GYROSET bestimmt, welches Servo einer Servogruppe von einem bestimmen Kreiselsensor kontrolliert wird. [Motor n der Gruppe] wird dabei für jeden Servo der Gruppe benutzt. Beachten Sie die folgenden Beispiele:

Beispiel des Befehls:
GYROSET G6B, 1, 1, 2, 2, 0, 0
Servomotor #6 empfängt die Daten von Gyro #1 und führt sie aus
Servomotor #7 empfängt die Daten von Gyro #1 und führt sie aus
Servomotor #8 empfängt die Daten von Gyro #2 und führt sie aus
Servomotor #9 empfängt die Daten von Gyro #2 und führt sie aus
Die Motoren #10 und #11 nutzen keinen Gyro-Sensor

GYROSENSE

Setzt jeden Servomotor, der am Gyro-Sensor angeschlossen ist

Befehlsstruktur:
GYROSENSE [Gruppe], [Gyroempfindlichkeit für Motor n] …

Erklärung des Befehls:
Die Anzahl der Gyros, die an eine Kontrolleinheit der MR-C3000-Serie angeschlossen werden können beträgt 4. Für jeden Gyro-Sensor kann man die Empfindlichkeit einstellen. Die [Gyroempfindlichkeit für Motor n] nutzt Numerale zwischen 0 und 255. Dabei reflektiert 0 den Wert des Gyros gar nicht auf den Motor.
Wenn der Wert größer ist erfolgt eine schnellere Reaktion auf Änderungen vom Wert des Gyro-Sensors.

Beispiel des Befehls:
GYROSENSE G6A, 100, 100, 255, 255, 50, 50
Die Servomotoren #0 und #1 nutzen eine Empfindlichkeit von 100
Die Servomotoren #2 und #3 nutzen eine Empfindlichkeit von 255 (Maximum)
Die Servomotoren #4 und #5 nutzen eine Empfindlichkeit von 50

Kapitel 12

roboBASIC

Prozessbefehle und sonstiges

ON … GOTO
Bedingungen mit Werten von Variablen vergleichen

Befehlsstruktur:
ON [Variable] GOTO [Sprungmarke], [Sprungmarke], …

Erklärung des Befehls:
Der Befehl ON … GOTO wird genutzt, wenn ein Programmprozess entsprechend dem Wert einer [Variable] geändert werden soll. Zusätzlich kann auch der IF-Befehl genutzt werden. Allerdings kann dieser dann auch durch ON … GOTO ersetzt werden. Im Vergleich zu IF kann außerdem ein kleinerer Code erstellt werden.

Das folgende Beispiel vergleicht IF mit ON … GOTO

```
A = BYTEIN(0)

IF A = 0 THEN
  GOTO 10
ELSEIF A = 1 THEN
  GOTO 20
ELSEIF A = 2 THEN
  GOTO 30
ELSE
  GOTO 40
ENDIF
```

```
A = BYTEIN(0)

ON A GOTO 10, 20, 30
IF A>2 THEN GOTO 40
```

Die [Sprungmarke] wird in Abhängigkeit von GOTO genutzt und hängt an dem Wert der Variable, der nach und nach um 1 steigt.
Numerale, Konstanten oder Variablen können für die [Variable] genutzt werden. Die maximale Größe der [Sprungmarke] ist 255.

RND
Zufall

Befehlsstruktur:
RND
Erklärung des Befehls:
Um mit Kontrolleinheiten der MR-C-Serie Zufallszahlen zu generieren wird der RND Befehl genutzt. Es werden Zufallszahlen zwischen 0 und 255 generiert.

Beispiel des Befehls:

```
DIM A AS BYTE

A = RND
BYTEOUT 0, A                    Ausgabe des Zufallswertes am Byte-Port 0
```

REMARK

Bemerkungen

Befehlsstruktur:
REMARK [Beschreibung]

Erklärung des Befehls:
Bei der Programmierung mit roboBASIC wird , oder REMARK (Beschreibung) genutzt um das Programm zu kommentieren.
Die [Beschreibung] sollte innerhalb einer Zeile bleiben. [Beschreibungen] beeinflussen die Programmablauf gar nicht!

Beispiel des Befehls:

```
REMARK 8 LEDs werden eingeschalten
BYTEOUT 0, 0
```

Kapitel 13

roboBASIC

Befehlserklärungen

$DEVICE

Stellt den Typ der Kontrolleinheit für das Programm ein

Befehlsstruktur:
,$DEVICE [Kontrolleinheit]

Erklärung des Befehls:
Dieser Befehl sorgt dafür, dass das Programm nach dem kompilieren auf die Kontrolleinheit geladen wird. Wenn eine bestimme [Kontrolleinheit] ausgewählt wurde wird das Programm spezifisch auf diese zugeschnitten.

Beispiel des Befehls:

```
,$DEVICE MRC2000                Programm für die Kontrolleinheit MR-C2000
```

,$LIMIT

Limitiert den Bewegungsbereich von Servomotoren

Befehlsstruktur:
,$LIMIT [Motornummer], [minimaler Wert], [maximaler Wert]

Erklärung des Befehls:
LIMIT dient dazu den Bewegungsbereich von Servomotoren einzuschränken um Schäden am Motor durch ein Überschreiten des möglichen Bewegungsraums zu vermeiden. [Motornummer] können Zahlen von 0 bis 31 sein. Der [minimale Wert] und der [maximale Wert] nutzen den Motorbewegungsbereich von 10 bis 190. Alle Motoren haben den Bewegungsbereich von 10 bis 190.

Beispiel des Befehls:

,$LIMIT 0, 50, 100	Der Bereich der Bewegung des Servomotors wird auf 50 – 100 beschränkt

Zubehör / Accessories

# 138018 1000mAh NiMh Battery / 6V (5Cell), 2/3A Size	
# 92528 Ladegerät MULTIcharger 810 WWC Charger MULTIcharge 810 WWC	
# 138020 Serial Interface Kabel Serial Interface Cabel	
# 138017 Anschlusskabel für HSR-8498HB (420mm) HSR-8498HB Connector Cable (420mm)	
# 138022 IR Sensor + Remote Control	
# 136050 Adapterkabel für Lader MULTIcharger 810 WWC	
# 138019 Quick Charger for 220V	

Robot-Servo

# 138006 HSR-8498HB	
# 113995 HSR-5995TG Robot	

Ersatzteile / Spare part

# 138014 Karbonite Getriebe Set für Servo HSR-8498HB HSR-8498HB Karbonite Gear Set	
# 138016 Servo Hebel und Schraubenset für HSR-8498HB HSR-8498HB Horn & Horn Screw Set	
# 138015 Oberes und unteres Gehäuse & Schrauben Set Top, Lower Cases & Screws Set	

Halterungen / Brackets

# 138011 Universal Halter **I**-Typ (2Stück für 1set) **I** Type Universal Bracket (2pcs for 1 set)	
# 138010 Universal Halter U Typ ohne Gewinde Non-tapped U Type Universal Bracket	
# 138009 Universal Halter U Typ mit Gewinde Tapped U Type Universal Bracket	
# 138013 Halter für vordere Schulter Shoulder Front Bracket	
# 138012 Halter für hintere Schulter Shoulder Back Bracket	

Bedienungsanleitung

ROBONOVA-I



MULTIPLEX Modellsport GmbH & Co. KG
www.robonova.de

Deutsch V1.00